PI Interface for Emerson DeltaV Batch

5.1.0

© 2015-2023 AVEVA Group Limited or its subsidiaries. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of AVEVA Group Limited. No liability is assumed with respect to the use of the information contained herein.

Although precaution has been taken in the preparation of this documentation, AVEVA assumes no responsibility for errors or omissions. The information in this documentation is subject to change without notice and does not represent a commitment on the part of AVEVA. The software described in this documentation is furnished under a license agreement. This software may be used or copied only in accordance with the terms of such license agreement.

AVEVA, the AVEVA logo and logotype, OSIsoft, the OSIsoft logo and logotype, ArchestrA, Avantis, Citect, DYNSIM, eDNA, EYESIM, InBatch, InduSoft, InStep, IntelaTrac, InTouch, Managed PI, OASyS, OSIsoft Advanced Services, OSIsoft Cloud Services, OSIsoft Connected Services, OSIsoft EDS, PIPEPHASE, PI ACE, PI Advanced Computing Engine, PI AF SDK, PI API, PI Asset Framework, PI Audit Viewer, PI Builder, PI Cloud Connect, PI Connectors, PI Data Archive, PI DataLink, PI DataLink Server, PI Developers Club, PI Integrator for Business Analytics, PI Interfaces, PI JDBC Driver, PI Manual Logger, PI Notifications, PI ODBC Driver, PI OLEDB Enterprise, PI OLEDB Provider, PI OPC DA Server, PI OPC HDA Server, PI ProcessBook, PI SDK, PI Server, PI Square, PI System, PI System Access, PI Vision, PI Visualization Suite, PI Web API, PI WebParts, PI Web Services, PRiSM, PRO/II, PROVISION, ROMeo, RLINK, RtReports, SIM4ME, SimCentral, SimSci, Skelta, SmartGlance, Spiral Software, WindowMaker, WindowViewer, and Wonderware are trademarks of AVEVA and/or its subsidiaries. All other brands may be trademarks of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the license agreement with AVEVA Group Limited or its subsidiaries and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12-212, FAR 52.227-19, or their successors, as applicable.

AVEVA Third Party Software Notices and Licenses: AVEVA Third Party Software Notices and Licenses | Legal | AVEVA

Contents

Introduction to PI Interface for Emerson DeltaV Batch v5.1.0 5
Interface modes 5
Supported features 7
Reference manuals 9

Batch interface configuration 10
Installation prerequisites 15
Security overview 16
Assign permissions for user accounts and PI points 17
Configure security for the PI Asset Framework 17
Configure security for PI Data Archive 18
Create and configure the interface instance 18
Batch interface failover 19
Configure interface instances for failover 22
PI server connectivity 23
Logging and error messages 23

How PI interfaces for batch and manufacturing execution systems work 25
How batch interfaces process event data 25
Equipment arbitration events unavailable 25
Interface modes 25
How batch data is stored 27
Batch list management 27
Event frames 28
Equipment hierarchy 31
Linking BES to MES batches using event frames 32
Placeholders and advanced parsing 34
Templates for mapping data source events 37
Tag templates 37
Tag template examples 40
Link templates 43
Property templates 44
Recipe templates 46
Recipe template examples 50
Recipe templates default properties 51
Filter incoming events 53
Translate foreign language data 53

PI Interface for Emerson DeltaV Batch 55

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Data sources . . . 55
Start and stop events . . . 57
Template placeholders . . . 62

PI Event Frames Interface Manager . . . 65
Server Information tab . . . 65
Windows Credential Manager for authentication . . . 66
Source tab . . . 67
Templates tab . . . 68
Alarm tag templates . . . 68
Triggers . . . 70
Property templates . . . 70
Recipe templates . . . 72
Recipe templates default properties . . . 74
Tag templates . . . 77
Filters tab . . . 78
Time Settings tab . . . 78
Batch Setup tab . . . 80
Operational Settings tab . . . 83
Monitor Tag Setup tab . . . 86
Test configuration tab . . . 89

Command line parameter reference . . . 90

Initialization file reference . . . 101
Source templates . . . 101
Recipe templates . . . 104
Tag templates . . . 105
Property templates . . . 111
Translation templates . . . 117
Initialization file examples . . . 117

Diagnostic point reference . . . 122

Glossary . . . 126

Release Notes . . . 130

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Introduction to PI Interface for Emerson DeltaV Batch v5.1.0

Emerson DeltaV is is a plant operations digital automation system that uses batch events to manage production processes.

PI Interface for Emerson DeltaV Batch is a scan-based interface that collects batch processing events from the DeltaV System, collecting batch events in real-time through the DeltaV OPC Alarm & Events Server (A&E Server). The interface stores them in PI Batch Database, and collects associated batch data to PI Tags and PI Batch properties. It populates the PI Batch Database and PIModule Database.

Associated batch data, such as operator comments, report parameters, and recipe parameters, are retrieved by querying the DeltaV Batch Historian during each interface scan. If you lose the connection to the DeltaV OPC A&E Server, the interface retrieves batch data and associated batch data from the DeltaV Batch Historian during each interface scan.

The interface automatically tries to re-establish the connection to the DeltaV OPC A&E Server; once the connection has been re-established, the interface returns to collecting batch data in real-time through the DeltaV OPC A&E Server.

This interface is primarily designed to be used for DeltaV 10.3 and later systems utilizing the DeltaV OPC A&E Server and the DeltaV Batch Historian; however, it can run against earlier systems utilizing different data sources.

- For DeltaV 9.3 systems, this interface can utilize the DeltaV Batch Historian or DeltaV event files as the primary data source.
- For DeltaV 8.4 systems, this interface can only use DeltaV event files as the primary data source.

NOTE: The use of DeltaV event files as a public interface for the DeltaV System is not recommended by Emerson.

The flow of data in the interface is unidirectional. Data can only be read from the specified data source and written to the PI Server. This interface can read data from multiple batch data sources simultaneously. By design, the interface does not edit or delete source data.

In addition to batch data, the interface can populate the PI Point Database. PI Point creation, commonly known as tag creation and event population, is controlled by using tag templates. All modules, tags, tag aliases, and health tags are automatically created on the PI server. The Interface does not use the PI API Buffering Service because batch and tag data is already buffered by the source historian databases. To maximize performance, the interface writes events to PI tags in bulk; that is, it writes all events per interface scan.

Interface modes

PI Interfaces for Batch and Manufacturing Execution Systems can be run in five different modes:

- RealTime (default)
- Recovery
- Preprocess
- Statistics

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- Delete

RealTime mode

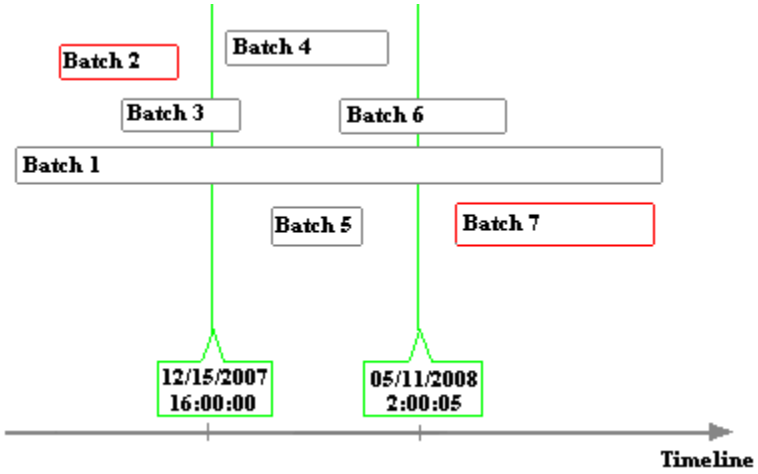
In **RealTime** mode, the interface monitors the data source for events that indicate the start or end of a batch or any child level thereof, recording these events in the PI System according to your batch configuration. The interface records newly-acquired data at the end of each scan regardless of whether batches are completed on the source. At startup, before it begins real-time data collection, the interfaces attempts to recover any data written after it was shut down.

Recovery mode

To recover events that occurred during interface downtime, the interface scans the data source for a specified period. If you omit an end time or specify * (current time), the interface recovers data and then starts collecting data in **RealTime** mode. If you specify an end time, the interface recovers data for the specified period and then exits. To ensure that history is recovered as completely as possibly, specify an end time of * (current time).

In recovery mode, the interface reads batch data from the data source for the specified time period. This mode can be used to initialize the PI System with historical data from the data source. If batch data for the specified period already exists in the PI System and the interface detects discrepancies, it attempts to correct the PI System data, logging any errors.

For example, the following figure shows a data source that contains batch data for seven batches.



If you recover data for the period from 12/15/2007 16:00:00 through 05/11/2008 2:00:05, the interface recovers contained batches (Batch 4 and 5) as well as border batches (Batch 1, Batch 3 and Batch 6). Batches outside the time frame (Batch 2 and 7) are not recovered.

Preprocess mode

If your data source contains data with timestamps that are earlier than the period covered by the primary archive, you can recover events by running the interface in *Preprocess** mode, which scans the data source and creates the required tags, modules and units in the PI System. After running the interface in **Preprocess** mode,

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

you must reprocess older archives to create entries for the tags, modules and units, then run the interface in recovery mode. This process is also referred to as "backfilling". (See the Backfill existing archives from new PI points for details on reprocessing archives.)

Note: For version 4 batch interfaces running with PI server version 390 or later, preprocessing is not required before recovering data.

Statistics mode

In **Statistics** mode, the interface scans a specified period, comparing data from the data source with the corresponding data in the PI Batch database. After comparing data, the interface reports the results and exits. If you omit an end time, the interface scans from the specified start time until the current time. To run the interface in Statistics mode using PI Event Frames Interface Manager, go to the **Operation Settings** tab, choose **Statistics** mode and specify the start and end times for the period to be analyzed. The interface logs results to the specified output file and exits.

Delete mode

In **Delete** mode the interface will delete batch data from the PI Batch Database or from event frames for a time period specified by start and end times. To run the interface in **Delete** mode using PI Event Frames Interface Manager, go to the **Operation Settings** tab, choose **Delete** mode and specify the start and end times for the period to be deleted. To delete data through the current time, omit the end time. If you intend to recover events from the data source for a specified time period, consider deleting existing batch data for that time period first.

Supported features

Platforms: 64-bit

- Windows 8.1
- Windows 10
- Windows 11
- Windows Server 2012
- Windows Server 2012 R2 SP1
- Windows Server 2016
- Windows Server 2019
- Windows Server 2022

This interface is exclusively built in 64-bit.

Feature	Support
Auto-creates PI Points and equipment assets?	Yes
Point Builder Utility	No
Stores batch data in PIBatch Database and PI Module Database	No
Stores batch data in PI AF	Yes
Supports equipment arbitration events	No

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Feature	Support
ICU Control	No (use PI Event Frame Interface Manager configuration tool)
PI Point Data Types	Int32/ Float32/ String
Sub-second Timestamps	Yes
Sub-second Scan Classes	No
Automatically Incorporates Changes to PI Point Attributes	No
Exception Reporting	No
Outputs from PI	No
Inputs to PI	Event and Scan-based
Supports Questionable Bit	No
Supports Multi-character Pointsource	Yes
Maximum Point Count	No maximum
Uses AF SDK	Yes: version 2.5.x or higher required
PINet String Support	N/A
Source of Timestamps	BES (not system time on interface node)
History Recovery	Yes
UniInt-based	No
Disconnected Startup	No
SetDeviceStatus	Yes
Failover	Yes
Vendor Software Required on PI Interface Node	No
Vendor Hardware Required	No
Additional PI Software Included with Interface	Yes
Device Point Types	The interface receives data from source as strings and coerces the data into numerical types according to tag templates, if defined.
Serial-Based Interface	No

Equipment arbitration events

This PI Batch Interface runs against a BES or MES that supports equipment arbitration events. These events provide precise time stamps for the start and end times of unit batches. Individual installations of BES or MES that support equipment arbitration events may not be configured to provide such events. If the BES or MES supports equipment arbitration events but is configured not to provide them, use the following command line parameter: /noarbitration.

Using `/noarbitration` ensures that the interface:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- Sets the start time of unit batches using the timestamp of the "Unit Procedure Started" event or the start of the next operation or phase for the unit, whichever is later.
- Sets the end time of unit batches using the timestamp of the "Unit Procedure Finished" event or the end of the last operation or phase for the unit, whichever is earlier.
- For operation-level recipes, the interface uses the start time of the first phase as the start time for the parent operation and unit batch, and the timestamp of the "Operation Finished" message as the end time of the phase, operation and unit batch.

History recovery

You can stop the interface without losing any data, because the data is persistent in the data source. Data recovery is limited by the history available from the BES, the number of licensed PI tags, and the size and time frame of the PI archives into which data is recovered.

Device status tag

This string tag contains information about communication between the interface and the data source. This tag is evaluated only while the heartbeat tag is updating and is updated on startup, change, and shutdown.

During normal operation, the tag contains the digital state set value "Good", indicating that the interface is communicating properly with the data source. Otherwise, the tag contains a string indicating status.

The following table lists standard status strings.

Message	Description
`1 Starting`	The interface is starting.
`2 Connected / No Data`	The interface is connected to the data source but is not capable or reading or writing data to the foreign device.
`3 n device(s) in error`	The interface is not able to communicate with the specified number of devices. Usually includes additional interface-specific details.
`4 Intf Shutdown`	The interface is shutting down.
`5 interface_specific_message`	Message specific to the interface.

Reference manuals

See the following user guides for more information on the operation of PI Interface for Emerson DeltaV Batch:

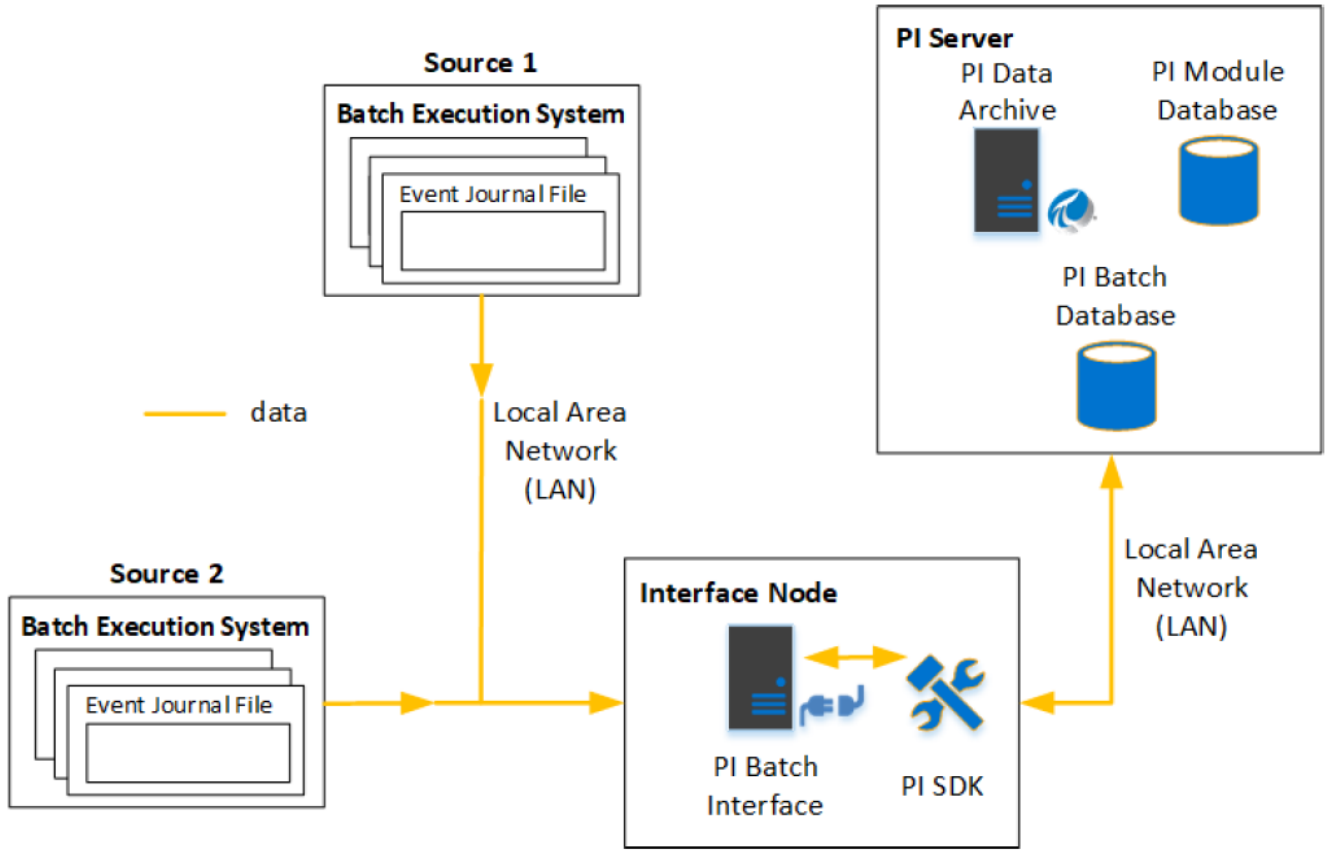
- PI Data Archive Reference
- PI System Management Tools
- AF SDK Reference

For details about the format and contents of source data, refer to the documentation provided by your batch execution system vendor.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

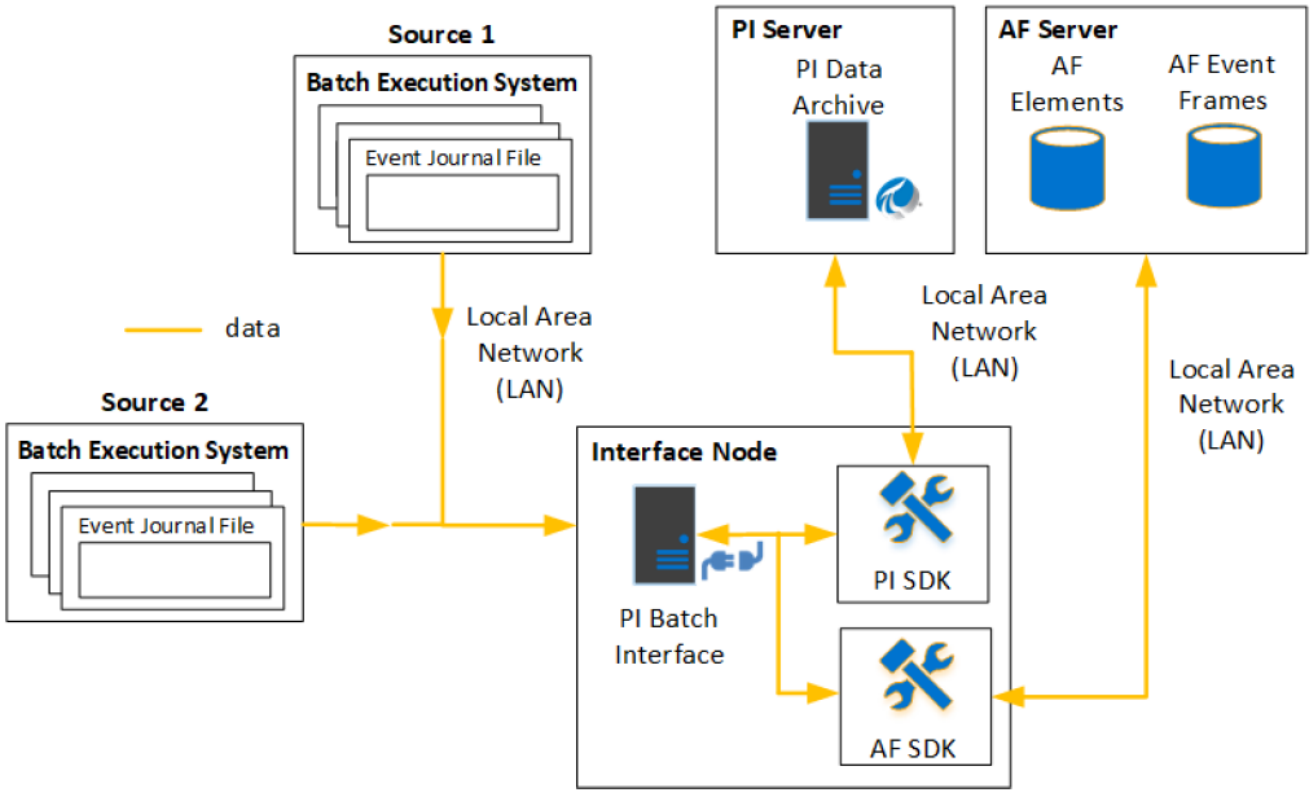
Batch interface configuration

The following diagrams illustrate various options for configuring batch interfaces, depending on the data source and the desired PI System output.



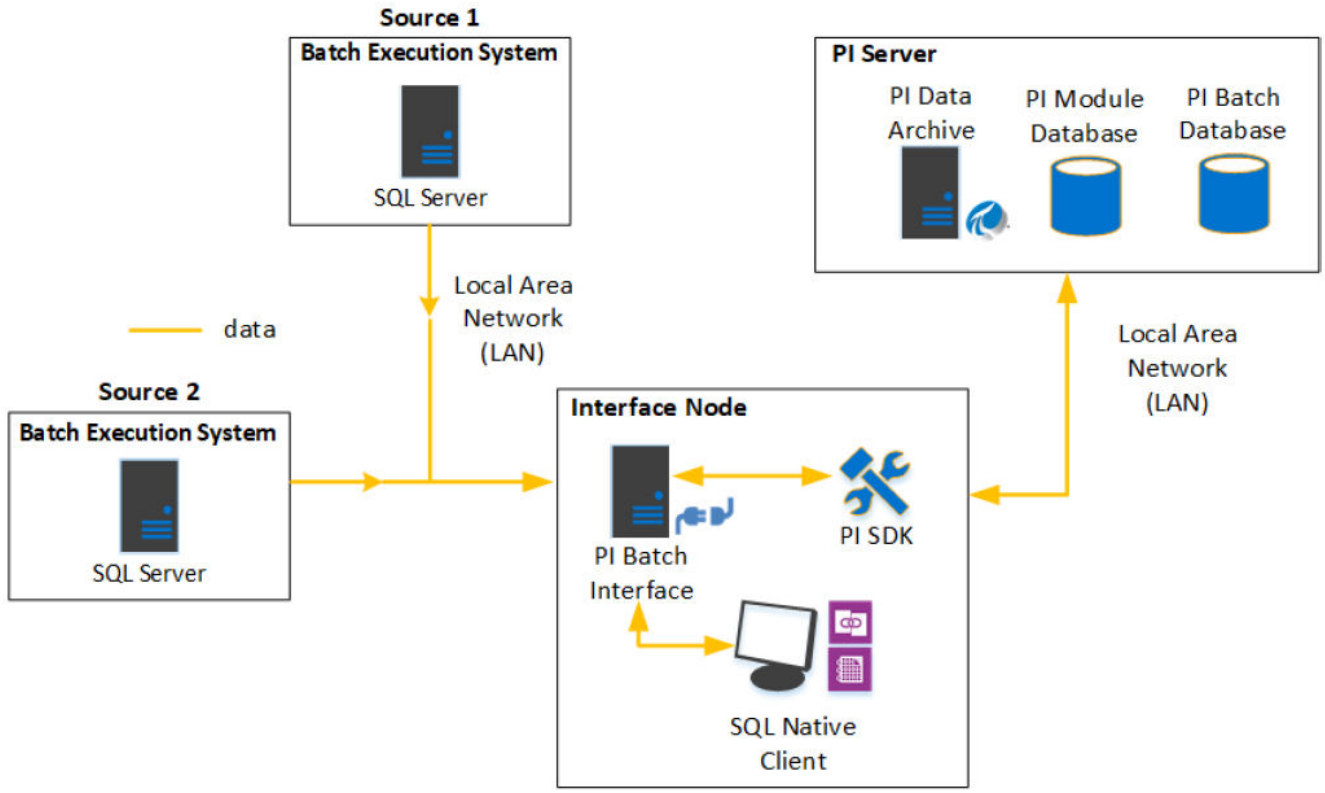
Data source: event files. Target: PI batch database

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



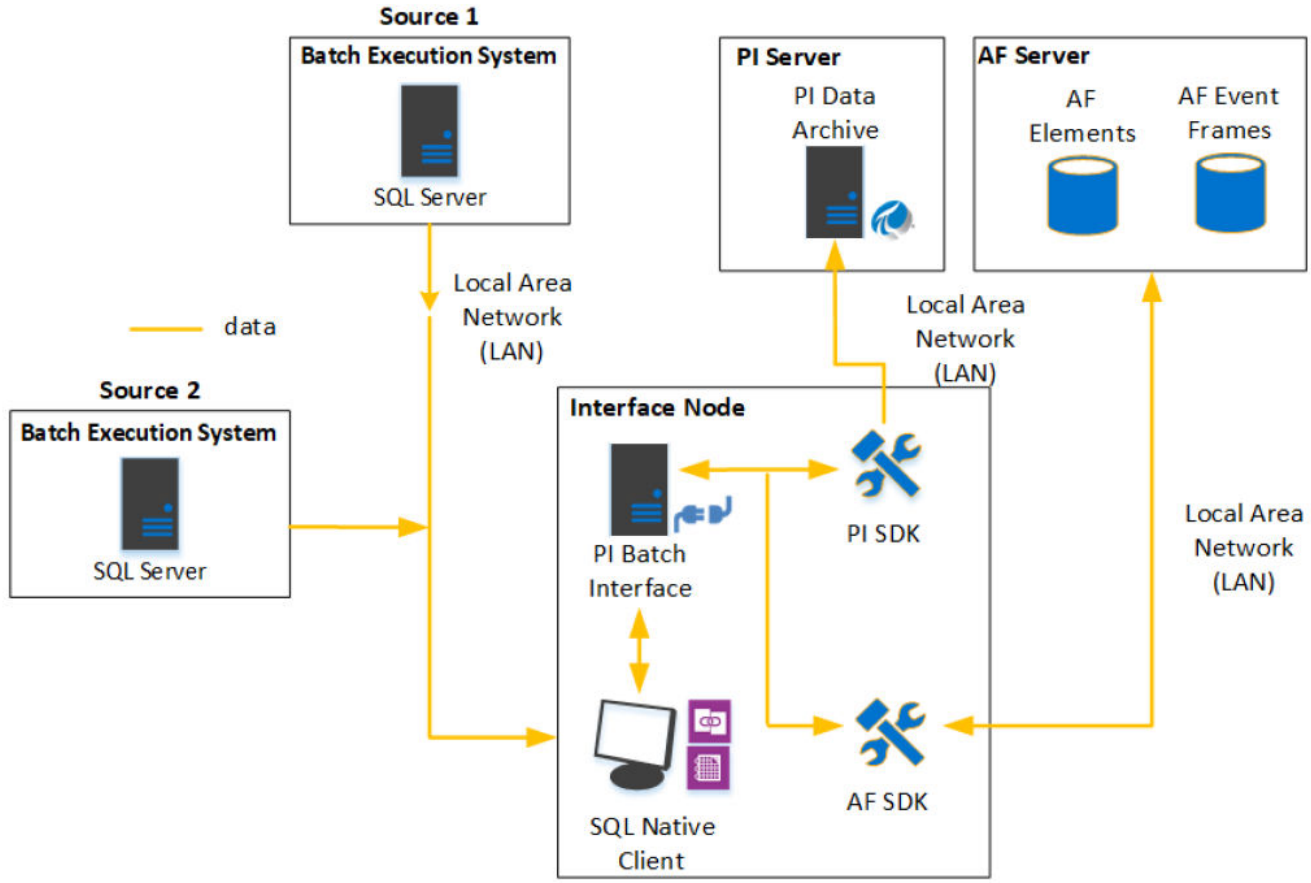
Data source: event files. Target: PI AF (event frames and assets)

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



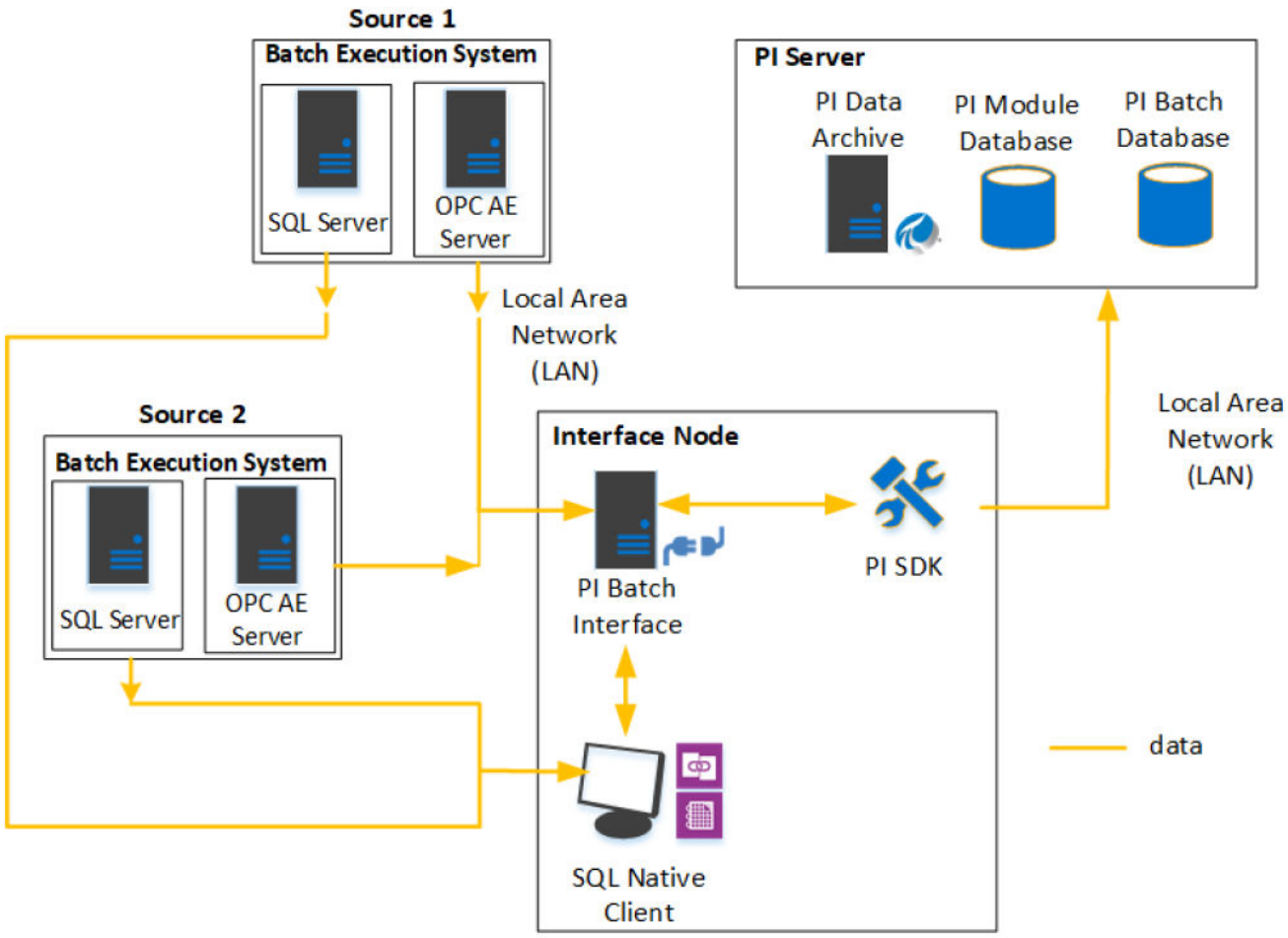
Data source: SQL Server. Target: PI batch database

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



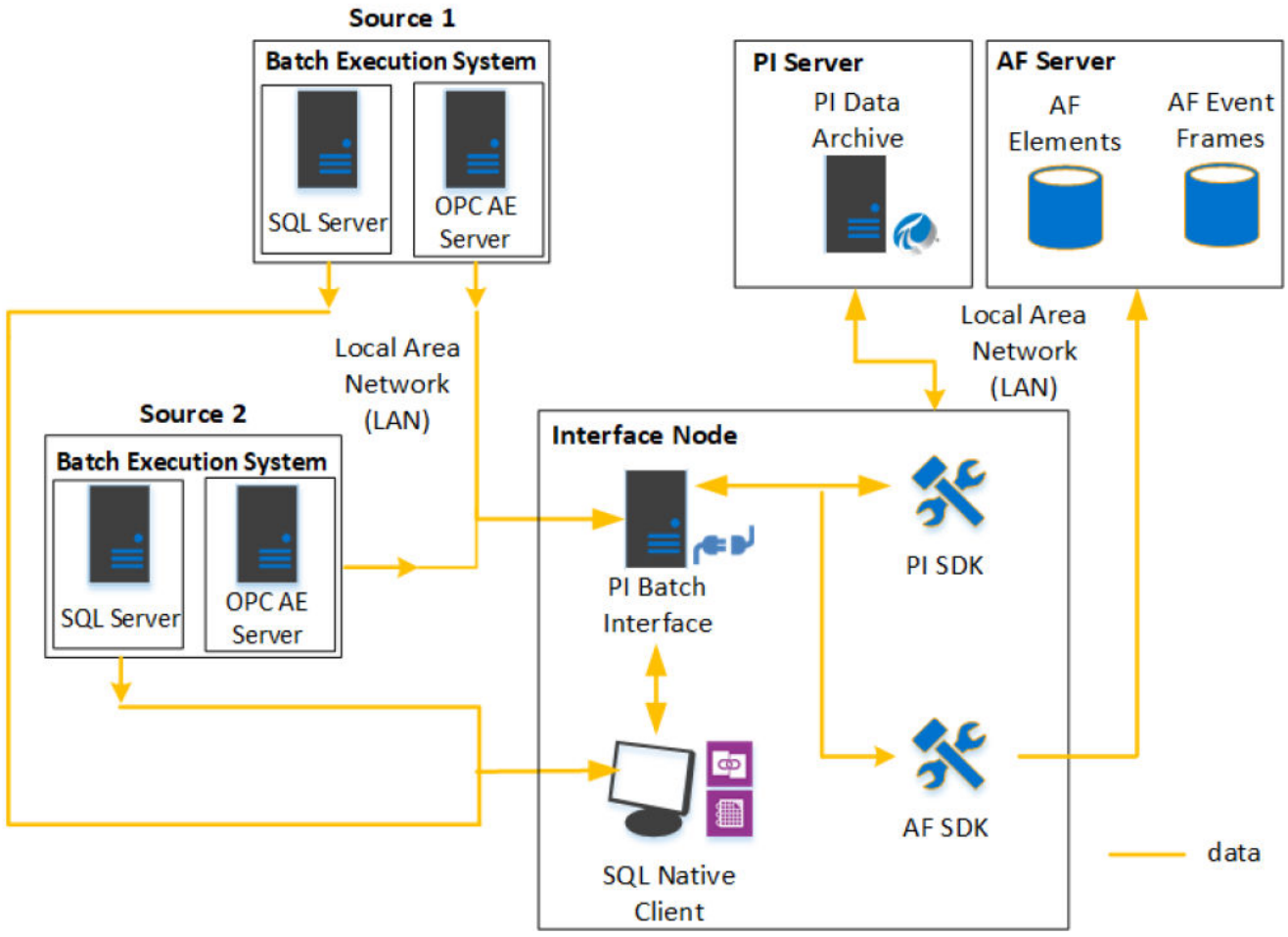
Data source: SQL Server. Target: PI AF (event frames and assets)

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



Data source: SQL Server + alarms and events server. Target: PI batch database

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



Data source: SQL Server + alarms and events server. Target: PI AF (event frames and assets)

Installation prerequisites

Minimum required versions of PI Server software: PI Server 2015 and PI AF 2016 (2.8.2).

Before installing and configuring, ensure that the following prerequisites are met:

- Verify that the PI Data Archive is running and is accessible from the computer where you intend to run the interface (the interface node).
- If you intended to generate event frames, make sure that the PI AF Server is running and is accessible from the interface node.
- Ensure that the system time on all these computers is correct.
- Verify that your batch execution system (BES) is up and running and that the data source is accessible from the interface node.

To install the interface, download and run its setup kit. By default, the interface is installed in an interface-specific folder in the following location:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

`%PIHOME%\Interfaces\`

The interface installation directory contains all the files and folders required to configure and run the interface, and includes example configurations.

The interface can run on the same computer as the BES or on a dedicated node. To avoid affecting the performance of the PI Data Archive, do not install the interface on the same computer as the PI Data Archive. When installing the interface, reserve the C: drive for the operating system, and install the interface on another drive.

If the data source is Microsoft SQL Server, you must install the Microsoft SQL Native Client on the interface node. You can download the client from the MSDN web site. If the data source is an Oracle database, you must install the corresponding version of Oracle Provider for OLE DB.

To configure the interface, use the PI Event Frame Manager configuration tool.

Note: To avoid potential interface configuration and startup issues, if you intend to run the interface as a Windows service and need to configure explicit login settings for your BES data sources, ensure that the account you configure for the service is the same account you use to run the configuration tool.

Security overview

If you are running PI Data Archive 3.4.380.36 or later, you can take advantage of its support for Windows Integrated Security by running the interface service using a Windows account that has the required permissions on the PI Data server. To configure Windows Integrated Security, use PI SMT to define a mapping that assigns a PI identity that has the required permissions to the user or user's group.

Caution: To avoid issues with the encryption of credentials be sure to use the same account for the interface service as you use in the PI Event Frames Interface Manager utility. If the interface service logon is not the same as the account that was used to save the credentials in the PI Event Frames Interface Manager utility, decryption of the credentials will fail and the service will exit. This affects credentials stored for login to the data source (e.g. SQL Login) and/or explicit login to AF.

For pre-3.4.380.36 versions of the PI Data Archive, you must create a PI trust for the user that runs the interface and configuration tool. Limit the trust to the hostname or IP address of the interface node and the application name (BIFConfig.exe for the PI Event Frames Interface Manager). Set the following permissions for the user that runs the interface:

- PI Data Archive permissions (**PI SMT**: Browse to **Security > Database Security**)
  - PIBATCH: read/write
  - PIMODULES: read/write
  - PIPOINT read/write
  - PIAFLINK: read (required for batch-to-event frames migration)
- Per-point security (**PI SMT**: Browse to **Points > Point Builder**)
  - Points created by the interface (using tag templates): Set both PtSecurity and DataSecurity to read/write
  - If using EFGEN: DataSecurity read access to the active points
- PI Asset Framework permissions
  - Database: read/write
  - Categories: read/write
  - Element: read/write
  - Element templates: read/write/read data/write data

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- Event frames: read data/write data

To configure batch interfaces to run as a service, the user who runs the PI Event Frames Interface Manager must be in the local Administrators group (administrative privilege is required to create services). The following permissions are required to run the PI Event Frame Manager to configure batch interfaces:

- PI Data Archive permissions
  - PIMODULES: read
  - PIPOINT: read
  - PIARCDATA: read (required for batch-to-event frames migration)
- PI Asset Framework permissions
  - Database: read
  - Categories: read
  - Elements: read
  - Element templates: read/write

Assign permissions for user accounts and PI points

The user that runs the interface requires read/write access to the PI Data Archive. To create event frames and write data to elements and attributes, the user must have permission to connect to the PI Asset Framework. To configure the interface, the user must be in the local Administrators group. Assign the required permissions to the user that runs the interface and all users who need to run the PI Event Frames Interface Manager configuration tool.

1. Launch PI System Management Tools, and click **Security > Database Security**.
2. For the user account under which the batch interfaces run, set the PIPOINT table to read and write permission.
3. For the user account under which the batch interfaces run,set the PIBACKUP table to read permission.
4. For PI points maintained by batch interfaces, set security as follows:
   a. Set the PtSecurity to read and write permission for any point that the interface creates.
   b. Set the DataSecurity to read and write permission for any point to which the interface writes data.

   You can set PI point permissions using PI SMT by choosing **Points > Point Builder**.

   You can change point settings in bulk using the PI SMT plug-in for Microsoft Excel.

Configure security for the PI Asset Framework

To configure batch interface security settings for the PI Asset Framework, perform the following steps.

1. Launch PI System Explorer.
2. Click **Database** on the toolbar. The Select Database window opens, listing existing databases.
3. Right-click the database where you intend to store the batch event frames, and then click **Security**.
4. Browse to the category and enter the required settings:
   a. Set **Database** to read/write.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

b. Set **Categories** to read.
c. Set **Element** to read/write.
d. Set **Element templates** to read.
e. Set **Event frames** to read/write.

Configure security for PI Data Archive

If you are running PI Data Archive 3.4.380.36 or later, you can take advantage of its support for Windows Integrated Security by running the interface service using a Windows account that has the required permissions. To use Windows Integrated Security, use PI System Management Tools to define a mapping that assigns the Windows user (or group) a PI identity that has the required permissions.

For pre-3.4.380.36 versions of the PI Data Archive, you must create a PI trust for the user who runs the interface and configuration tool. For tightest security, limit the trust to the hostname or IP address of the interface node and the application name.

If you are installing the interface on a node other than the PI Data server, you must create trusts to ensure that the configuration tool and the interface have access to the server.

1. To create a trust, launch PI System Management Tools and connect to the target server.
2. Click **Security**, and then click **Mappings & Trusts**.
3. Right-click within the **Trusts** tab, and click **New Trust**.

   The **Add Trust** wizard launches.
4. Enter a meaningful name and description for the trust.
5. Configure the following settings:

Program	Type of Trust	Application Name
PI Event Frame Interface Manager	PI-API application	`BIFConfig.exe`
Interface executable	PI-SDK application	*Executable name*

Create and configure the interface instance

For each instance you create, settings are stored in a separate Windows command (.bat) file and an initialization (.ini) file in the `%ProgramData%\OSIsoft\Interfaces\EMDVB` directory. The batch file launches the interface, specifying settings as command line parameters. The initialization file also contains settings, and it defines templates that determine how data from the data source is stored in the PI System. To configure settings for interfaces, use the PI Event Frames Interface Manager. Use this tool even if you are configuring the interface to store data in the PI Batch Database rather than event frames.

A single batch interface instance can be configured to read from multiple data sources. This capability enables you to handle distributed batch processing scenarios, where multiple batch execution systems cooperate in the manufacturing of a single batch. If you configure multiple data sources, all data sources must be the same type, either event files or SQL databases.

Note: Each PI Interface for Emerson DeltaV Batch instance must run under a unique service account. For additional information, see Can I use the same service account to run multiple instances of EF-generating interfaces?.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

To launch the PI Event Frames Interface Manager, click **Start > All Programs > PI System > PI Event Frames Interface Manager**. For detailed information about the settings on each tab, see PI Event Frames Interface Manager.

To create an instance of the interface, perform the following steps using PI Event Frames Interface Manager:

1. On the **Interface Selection** tab, click **Add Interface**. A list of installed interfaces displays.
2. Choose the interface that you want to configure, and specify a service ID number and description if desired.
3. Click **OK**.
4. On the **Server Information** tab, specify settings for your PI servers.

   If you intend to create event frames, check **Create event frames** and specify the PI Asset server and AF database.
5. On the **Source** tab, configure the settings for the data source (the BES).

   Note that you can configure multiple data sources for the same interface instance.
6. On the **Templates** tab, define templates for tags, properties and the batch recipe.

   For details, see Templates for mapping data source events.
7. On the **Filters** tab, specify any batch levels from the data source that you do not want the interface to process.
8. On the **Time Settings** tab, configure retry and timeout settings.
9. On the **Batch Setup** tab, configure settings according to the requirements of your batch execution system.
10. On the **Operational Settings** tab, configure settings to determine how batch data is recorded, and any interface-specific settings required.
11. To save your changes, click **Save Settings**.

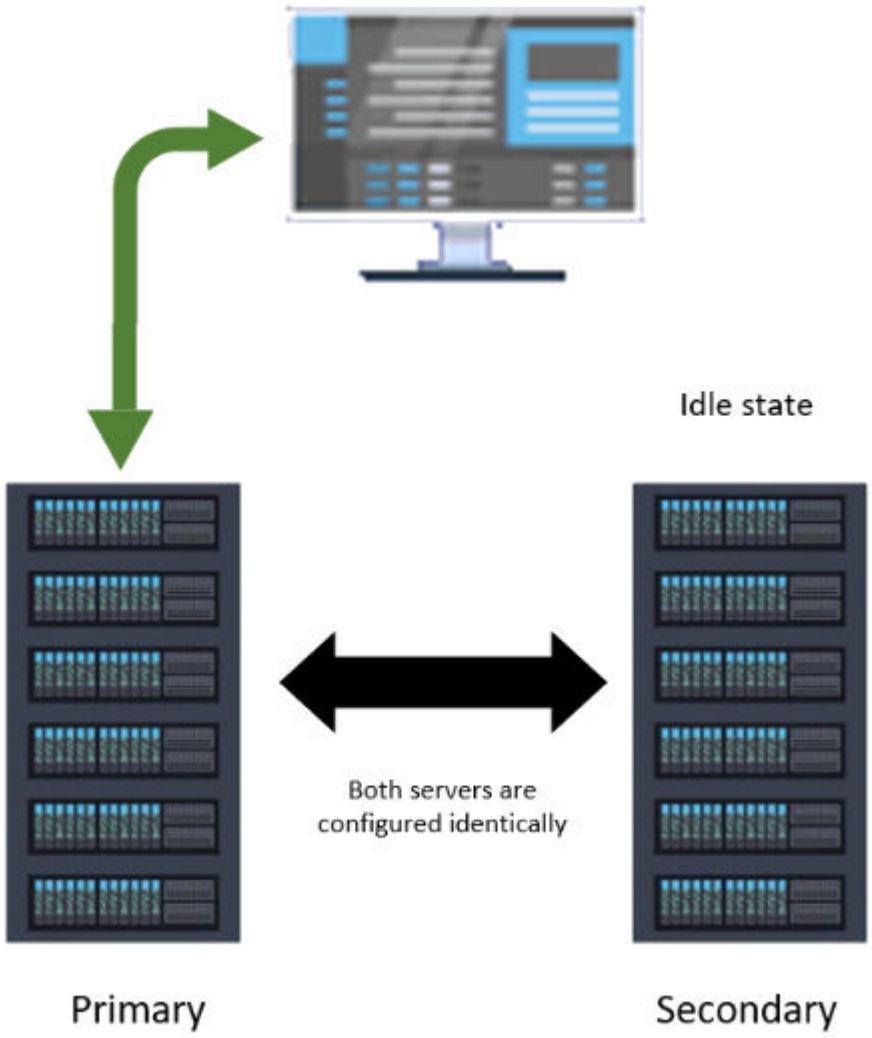
Batch interface failover

With interface failover, you can configure multiple interface instances for PI Interface for Emerson DeltaV batch to help minimize data loss in case of an unexpected interface machine shutdown. If the primary interface instance fails, then the secondary instance takes over and continues the data collection process.

For details on configuring failover for the interface, see Configuring interface instances for failover

The following example illustrates a typical interface failover set-up.

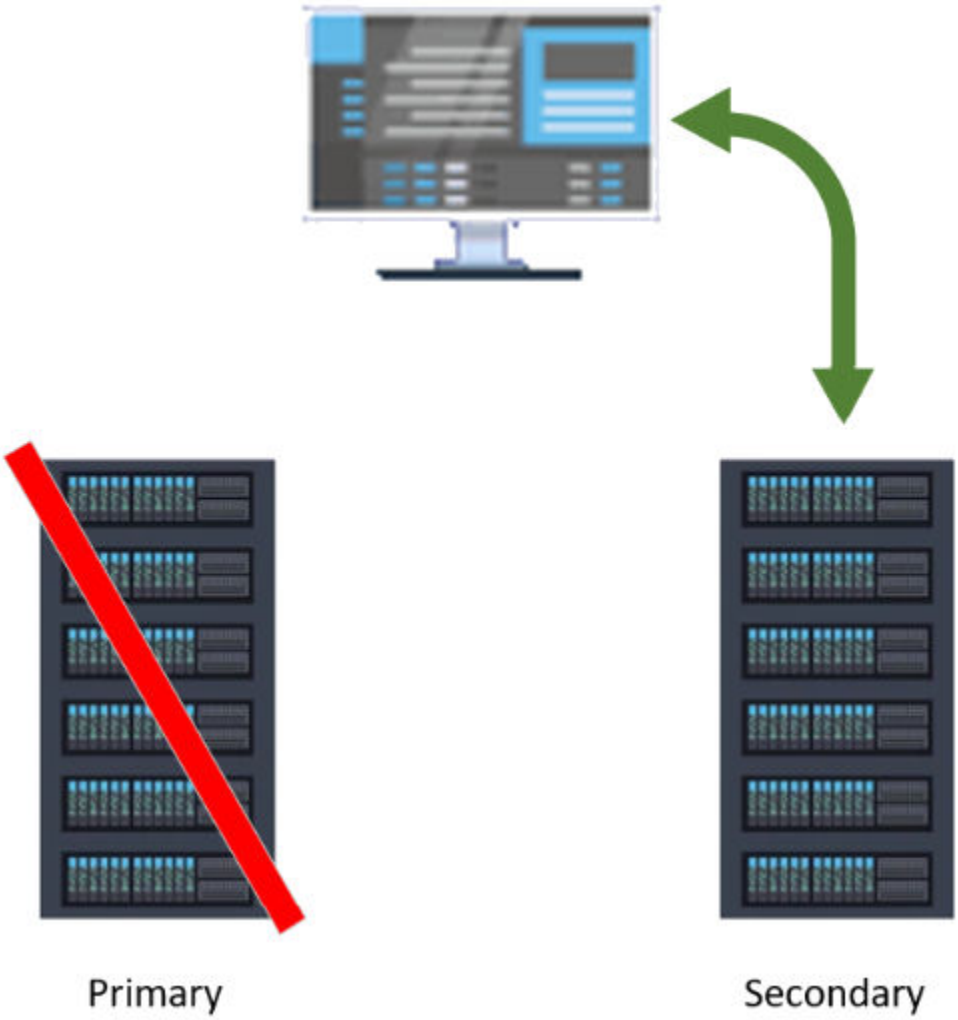
©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



The primary interface updates the failover tag with the timestamp of the last value written.

The secondary interface monitors the failover tag until the primary interface becomes inactive. The secondary instance takes over data collection and becomes the primary when the primary fails and the amount of time it is unavailable without the failover tag being updated exceeds the amount of time configured in the Failover swap time parameter.

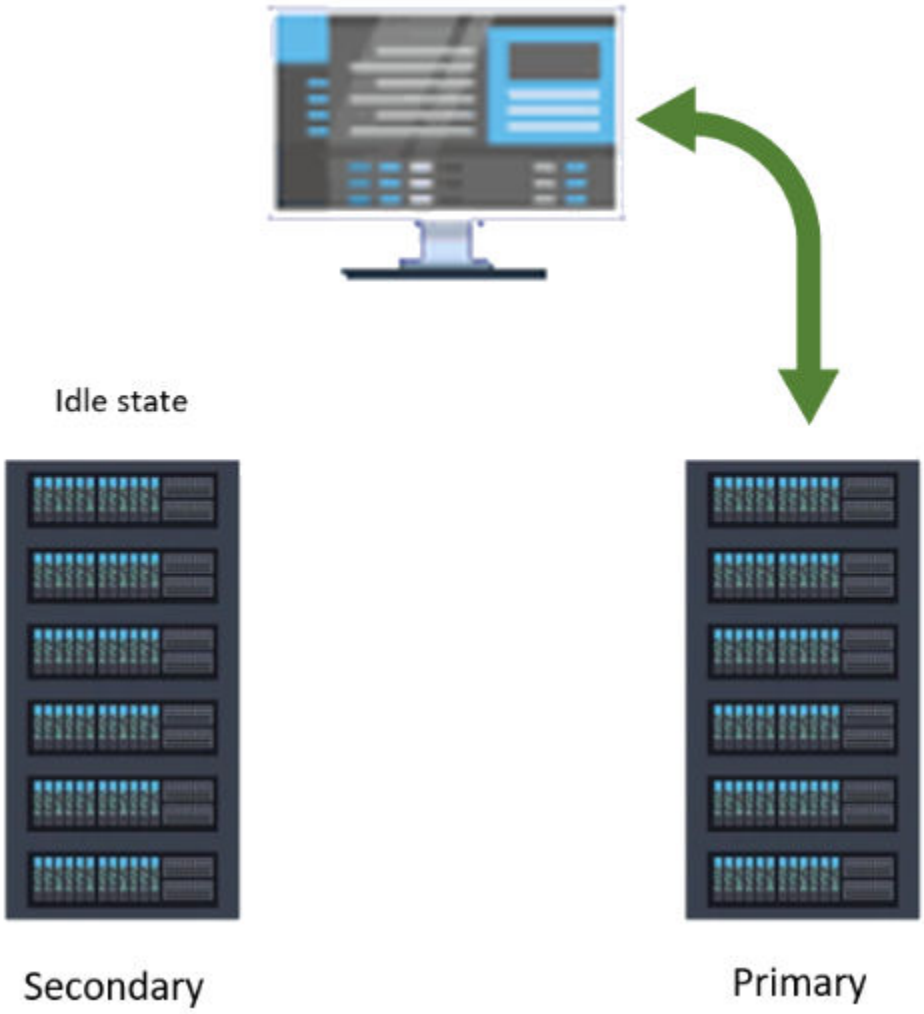
©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



It is important to note that when the secondary interface takes over the data collection, it does not pick up immediately where the primary stopped. Data is not duplicated. The secondary interface begins data collection based on the value configured in the Recovery start time (/rst parameter.

When the previously shutdown interface re-establishes connection, it goes into an idle state and become the secondary interface. The interface that was promoted to primary continues regular data collection operation.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



Configure interface instances for failover

To ensure that batch data continues to be collected if an instance of the interface fails, you can configure multiple interface instances to run in failover mode. The instances must be configured identically.

To enable failover, perform the following steps before you start the interface instances:

1. Configure instances of the interface on different host computers. Assign the same interface ID and point source, and assign unique failover IDs to each instance.

   You can configure more than two instances for failover.
2. In the target PI Data Archive, create a string tag and configure it with the same point source and Location1 (interface ID) as the interfaces.

   The interface instance uses this string tag to coordinate failover.
3. Using PI Event Frame Interface Manager, go to the **Operational Settings** tab and configure the failover settings.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- **Failover tag**

  The name of the tag that you created in the previous step.

- **Failover identifier**

  A unique ID for the interface instance.

- **Failover swap time**

  The amount of time that the current primary interface must be unavailable before failover occurs.

4. Start the interface instances.

   When the primary interface instance is operational, it updates the failover tag with the timestamp of the last value written. The backup instance monitors this tag and, if the failover swap time elapses and the failover tag has not been updated, the backup instance takes over data collection and becomes the primary instance.

PI server connectivity

If a connection is lost during processing, the interface suspends all actions until it reconnects to the PI Data server or the data source. If the data source connection is down, the interface tries to reconnect on every scan until it succeeds. If the PI Data server connection is lost, the interface attempts to reconnect periodically until it times out.

You can configure both the retry interval and the timeout period on the **Time Settings** tab of the PI Event Frames Interface Manager. The interface logs any connection errors that occur.

Note: Sending tag values to multiple servers requires configuring PI Buffering on the interface node. To configure the buffering services, refer to the Buffering section of the AVEVA PI Server guide.

Additionaly, creating tags and other configuration updates to multiple servers requires all servers to be part of a PI Server Collective. For additional information on how to configure collectives, refer to PI Data Archive collective management in the AVEVA PI Server guide.

Logging and error messages

The interface logs operational messages during interface startup, data collection and recovery. Additional messages are logged if you enable debugging.

To view messages, open the Windows Event Viewer on the interface's installed machine and go to the Applications and Services Logs. Locate the log using the generic interface name. The management of the logs can be done through the Windows Event Viewer.

Streaming messages for a batch interface display in the following format:

```
{timestamp:yyyy-MM-dd HH:mm:ss.fff} {logLevel,-11} {_iD}: {msg}
```

where {_iD} is the Interface ID instance.

Within the Windows Event Viewer, the Event ID displayed for each batch interface log is the Interface ID instance.

To enable debug output for troubleshooting, launch PI Event Frames Interface Manager, select your interface instance, and go to the Operational Settings tab.

Operational settings

Local debug messages (/DB=<#>)

Specifies level of detail for logging as follows:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- 0: Log errors and warnings (default)
- 1: Log errors, warnings and major successes
- 2: Log all messages
- 3: Log all messages including Unirecord diagnostic message

Command line parameter reference

/db =[#]

(Optional) Enabled debugging output:

- 0: Log only errors and warnings (default)
- 1: Log errors, warnings and major success messages
- 2: Log all messages (most verbose
- 3: Log all messages including Unirecord diagnostic message

Log properties

You can create the Source in the Windows Event Viewer by using the batch interface's source/log name (*see Source/log chart below*) before initially starting the interface. After starting the interface, the Windows Event Viewer creates the Source automatically.

Note: If you do not have appropriate permissions to create an event log, it is recommended that you run the interface interactively with an account that has permissions to create an event log.

To change the log path, increase or decrease the maximum size of the log or modify how the log behaves after reaching the defined maximum event log size, open the Windows Event Viewer and locate the source/log name under Applications and Services Log. Right-click on the source/log name and select **Properties**.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

How PI interfaces for batch and manufacturing execution systems work

Batch interfaces scan a data source for events of interest, such as the start or end of a level, and the acquisition and release of equipment.

Based on these events, the interface generates entries in the PI Batch Database or event frames. To handle configurations in which multiple batch execution systems manage related batch processes that you want to merge, you can configure a single interface instance to read multiple data sources.

How batch interfaces process event data

The interface processes start and end events for each level. The level at which a recipe executes depends on the equipment it requires. For example, a batch-level recipe is most likely composed of unit procedures and procedures executed on multiple different units. By contrast, an operation-level recipe might execute a set of phases in a single unit. The interface automatically creates PIBatches (or level 1 events) and PIUnitBatches (or level 2 events) for operation- and lower-level recipes, even though the events in the data source do not include these levels. The BES events that trigger the start and end of each level are vendor-specific. For details, refer to the vendor-specific information in this document.

When you configure the interface to generate event frames in PI AF, the interface creates a set of event frame templates in the target database, one template for each level in the standard S88 batch hierarchy. You can modify the templates to customize the data that is stored in the generated event frames. The interface creates equipment assets in the Module Database or PI AF (depending on where you're storing batch data) based on allocation events from the BES, and populates the attributes of those assets with relevant data.

Equipment arbitration events unavailable

If the batch execution system does not generate equipment arbitration events, select the "Disable Arbitration" option (/noarbitration). Note that, without equipment arbitration events, the interface cannot precisely determine the start and end of unit batches.

If you disable arbitration, the interface determines the start time of unit batches using the timestamp of the "Unit Procedure Started" event or the start of the next operation or phase for the unit, whichever is later. For end time, the interface uses the timestamp from the "Unit Procedure Finished" event or the end of the last operation or phase for the unit, whichever is earlier.

For operation-level recipes, the interface uses the start time of the first phase as the start time for the parent operation and unit batch, and the timestamp of the "Operation Finished" message as the end time of the phase, operation and unit batch.

Interface modes

PI Interfaces for Batch and Manufacturing Execution Systems can be run in five different modes:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- RealTime (default)
- Recovery
- Preprocess
- Statistics
- Delete

RealTime mode

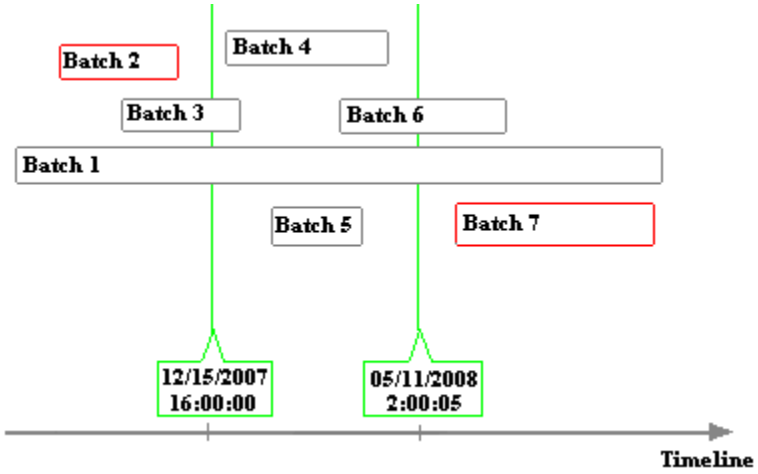
In **RealTime** mode, the interface monitors the data source for events that indicate the start or end of a batch or any child level thereof, recording these events in the PI System according to your batch configuration. The interface records newly-acquired data at the end of each scan regardless of whether batches are completed on the source. At startup, before it begins real-time data collection, the interfaces attempts to recover any data written after it was shut down.

Recovery mode

To recover events that occurred during interface downtime, the interface scans the data source for a specified period. If you omit an end time or specify * (current time), the interface recovers data and then starts collecting data in **RealTime** mode. If you specify an end time, the interface recovers data for the specified period and then exits. To ensure that history is recovered as completely as possibly, specify an end time of * (current time).

In recovery mode, the interface reads batch data from the data source for the specified time period. This mode can be used to initialize the PI System with historical data from the data source. If batch data for the specified period already exists in the PI System and the interface detects discrepancies, it attempts to correct the PI System data, logging any errors.

For example, the following figure shows a data source that contains batch data for seven batches.



If you recover data for the period from 12/15/2007 16:00:00 through 05/11/2008 2:00:05, the interface recovers contained batches (Batch 4 and 5) as well as border batches (Batch 1, Batch 3 and Batch 6). Batches outside the time frame (Batch 2 and 7) are not recovered.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Preprocess mode

If your data source contains data with timestamps that are earlier than the period covered by the primary archive, you can recover events by running the interface in *Preprocess** mode, which scans the data source and creates the required tags, modules and units in the PI System. After running the interface in **Preprocess** mode, you must reprocess older archives to create entries for the tags, modules and units, then run the interface in recovery mode. This process is also referred to as "backfilling". (See the Backfill existing archives from new PI points for details on reprocessing archives.)

Note: For version 4 batch interfaces running with PI server version 390 or later, preprocessing is not required before recovering data.

Statistics mode

In **Statistics** mode, the interface scans a specified period, comparing data from the data source with the corresponding data in the PI Batch database. After comparing data, the interface reports the results and exits. If you omit an end time, the interface scans from the specified start time until the current time. To run the interface in Statistics mode using PI Event Frames Interface Manager, go to the **Operation Settings** tab, choose **Statistics** mode and specify the start and end times for the period to be analyzed. The interface logs results to the specified output file and exits.

Delete mode

In **Delete** mode the interface will delete batch data from the PI Batch Database or from event frames for a time period specified by start and end times. To run the interface in **Delete** mode using PI Event Frames Interface Manager, go to the **Operation Settings** tab, choose **Delete** mode and specify the start and end times for the period to be deleted. To delete data through the current time, omit the end time. If you intend to recover events from the data source for a specified time period, consider deleting existing batch data for that time period first.

How batch data is stored

Batch interfaces support two options for storing batch data: as PI batches or as PI AF event frames. For PI batches, the interface represents equipment (units) by creating PI modules. For event frames, the interface represents equipment by creating PI AF assets. The batch database is a special-purpose database, optimized for (and constrained by) S88 and similar models, whereas PI AF event frames and elements provide a highly flexible, general-purpose structure that enables you to define your asset model and event hierarchy as needed. Most notably, the batch database imposes a one-unit-batch-per-unit restriction, whereas event frames do not restrict the number of unit batches that can be active in the same unit at the same time.

Batch list management

In real-time mode, the interface first checks to see if there are batches in the batch list. If there are no batches in the batch list after querying between the query start and end time, the start time remains the same for the next query.

If there are only closed batches in the list, the interface selects the batch with the most recent start time for the Next Query Start Time. If that time falls after the Current Query Start Time, it will be used as the Next Query

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Start Time. If the time does not fall after the Current Query Start Time, the Next Query Start Time remains the same.

If there are open batches in the list, the interface selects the batch with the most recent start time for the Next Query Start Time from the list of open batches. If that time falls after the Current Query Start Time, it will be used as the Next Query Start Time. If the time does not fall after the Current Query Start Time, the Next Query Start Time remains the same.

In history recovery mode, the configuration of the Maximum query time frame (/MAXQTF=) on the Time Settings tab determines the query end time. For example, if the MAXQTF is set to five days, the end time for the queries moves forward five days from the query start time. When the interface defines a query start time, the end time moves forward automatically based on the MAXQTF configuration.

Event frames

Use event frames to record batch data. Event frames allow you to use recipe templates to define the attributes to be stored in event frames at every level of the hierarchy.

Each event frame contains the following fields:

- Name
- Description
- Start time
- End time
- Template
- Category (Default is "OSIBatch")
- Batch properties (stored in event frame attributes)
- Referenced elements such as unit or phase module

To view batch event frames and AF elements generated by the interface, use PI System Explorer. In the root-level event frame, the Name field contains the batch ID from the data source. In lower-level event frames, the Name field contains the recipe name. In event frames for a procedure-level recipe, the product and recipe properties from the data source are stored as attributes. For unit procedure-level recipes, the batch ID and product from the data source are stored as attributes.

If the highest recipe level is an operation or phase (that is, neither procedure nor unit procedure levels are defined), the interface creates event frames that correspond to the procedure and unit procedure level.

The following sections explain how data is stored in event frames for each level of the batch hierarchy.

Procedure

For each batch in the data source, the interface creates a root-level event frame that represents the procedure in the recipe. Each root-level event frame contains a collection of child event frames that correspond to unit procedures. Data from source batch events (that is, properties) can be recorded in the attributes of the event frame and in PI points. Source batches can have identical IDs and recipe names in the same time frame.

To match the source batch with an event frame, the interface stores additional information in the extended properties of the root event frame. The Name property contains the ID of the source batch, and the Value property contains an XML structure composed of the following batch data:

- Batch ID

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- Product (searchable)
- Formula name
- Recipe (searchable)
- Recipe type
- Start time UTC
- End time UTC
- Interface name
- Interface ID
- Data source

The following table shows how the source batch properties map to event frame data.

Source Procedure Property	Event Frame Field	Event Frame Attribute
BatchID	Name	
Procedure Name		Recipe
Product		Product
Start Time	Start Time	
End Time	End Time	

By default, the interface captures the following batch-associated properties and stores them in procedure-level event frame attributes:

- Recipe header
- Formula header
- Recipe value
- Report

For the preceding attributes, the name is assigned from the source [DESCRIPTION] column, the value from the source [PVALUE] columns, and the unit of measure attribute from the [EU] column.

The procedure-level event frame can represent merged source batches. The product and recipe attributes contain data associated with the source batch that started the merged event frame. For each merged source batch, the interface creates an entry node in the event frame's extended properties, named using the unique ID of the source batch, with its value containing an XML structure composed of the original source batch properties.

Unit Procedure

The interface creates a unit procedure-level event frame for each unit procedure read from the data source. Each unit procedure-level event frame is a child of the procedure-level event frame and contains the subset of event frames that represent the source batch operation-level recipe. The start and end times of an event frame record the start and end of physical processing in a unit.

The name field of the unit procedure-level event frames contains the unit procedure name as read from the data source. The batch ID and product properties are searchable attributes of the event frame. The following table shows how, by default, the source batch properties map to event frame data:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Source UnitProcedure Property	Event Frame Field	Event Frame Attribute	Reference Element
BatchID		BatchID	
UnitProcedure Name	Name	Procedure	
Product		Product	
Start Time	Start Time		
End Time	End Time		
Unit			Unit

In addition to batch ID, procedure and product attributes, the interface records recipe and report events in event frame attributes. For these events, the attribute name is assigned from the [DESCRIPTION] column, the value from the source [PVALUE] column, and units of measure from the [EU] column.

Unit procedure-level event frame properties do not change if the parent object is a merged event frame. By default, unit procedure event frames contain the batch ID and procedure name read from the data source. (To override the default, use PI Event Frame Interface Manager to configure the Batch ID mask field on the Batch Settings page.)

When operation or phase-level recipes are run, the interface uses the operation or phase name as the name of the unit procedure-level event frame.

Operation

The interface creates an operation-level event frame for each operation read from the data source. Each operation-level event frame is a child of the unit procedure-level event frame and contains the subset of event frames that represent the source batch phase-level recipe.

The name field of the operation-level event frames is the operation name read from the data source. The following table shows how the source batch properties map to event frame attributes:

Source Operation Property	Event Frame Field	Reference Element
Operation Name	Name	
Start Time	Start Time	
End Time	End Time	
Unit		Unit

By default, the interface records recipe value and report events in event frame attributes. For these events, the attribute name is assigned from the [DESCRIPTION] column, the value from the source [PVALUE] column, and units of measure from the [EU] column.

Phase

The interface creates a phase-level event frame for each phase read from the data source. Each phase-level event frame is created as a child of an operation-level event frame and contains the subset of event frames that represent the source batch phase states-level recipe.

The name field of the phase-level event frame contains the phase name read from the data source. The following table shows how the source batch properties map to event frame attributes:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Source Phase Property	Event Frame Field	Reference Elements
Phase Name	Name	
Start Time	Start Time	
End Time	End Time	
Unit		Unit
Phase Module		Phase Module

By default, the interface records recipe value and report events in event frame attributes. For these events, the attribute name is assigned from the [DESCRIPTION] column, the value from the source [PVALUE] column, and units of measure from the [EU] column.

Phase State

The interface creates a phase state-level event frame for each phase state read from the data source. Each phase state-level event frame is created as a child of a phase-level event frame and, if so configured, can contain the subset of event frames that represent the phase steps. The start of new phase state ends the previous one, unless the new state is COMPLETE, ABORTED or STOPPED, which end the current phase state without beginning a new one. These phase states have a zero-duration time frame.

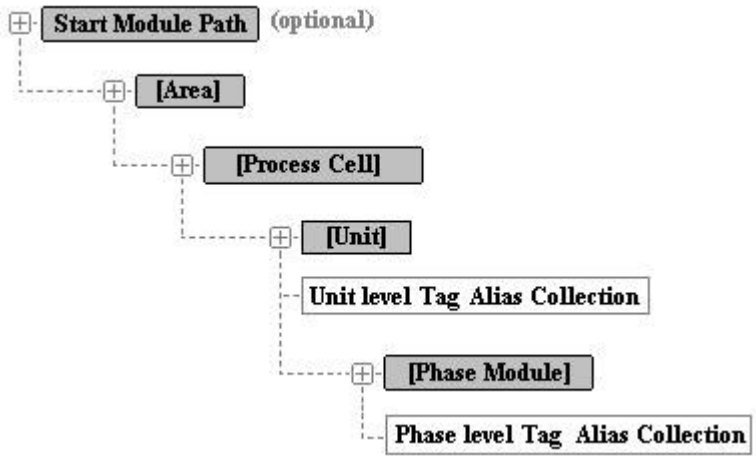
The name field of the phase state event frames reflects an actual source recipe phase state name. Below is the mapping of source phase state to event frame fields and attributes:

Source Phase State Property	Event Frame Field	Referenced Element
Phase State	Name	
Start Time	Start Time	
End Time	End Time	
Unit		Unit
Operation Module		Operation Module
Phase Module	Phase Module	Phase Step

Equipment hierarchy

When recording batch data from the data source in the PI Batch Database, the interface creates PI modules or PI AF elements representing equipment, according to the data from the data source. The following equipment hierarchy is created:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



By default, the hierarchy is located at the root level of the PI Module Database or the PI AF element hierarchy. To specify a different root, use PI Event Frame Interface Manager to configure the start module path.

Aliases associate unit with PI tags that track measurements of interest that are related to the unit, for example, temperature. By default, aliases are created at the unit and phase module level. Unit aliases are created for a tag if the template includes the **UNIT** placeholder, and phase aliases are created if it contains the **PHASEMODULE** placeholder. Unit aliases are also created if the tag name contains the unit name (for example, if the unit name is UNIT202 and the tag name is "UNIT202 report tag"), a unit alias is created; likewise for phase modules. All batch interfaces support unit- and phase-level aliases, and some support creation of aliases at any level of the batch hierarchy. For details, refer to the interface-specific section of this guide.

Linking BES to MES batches using event frames

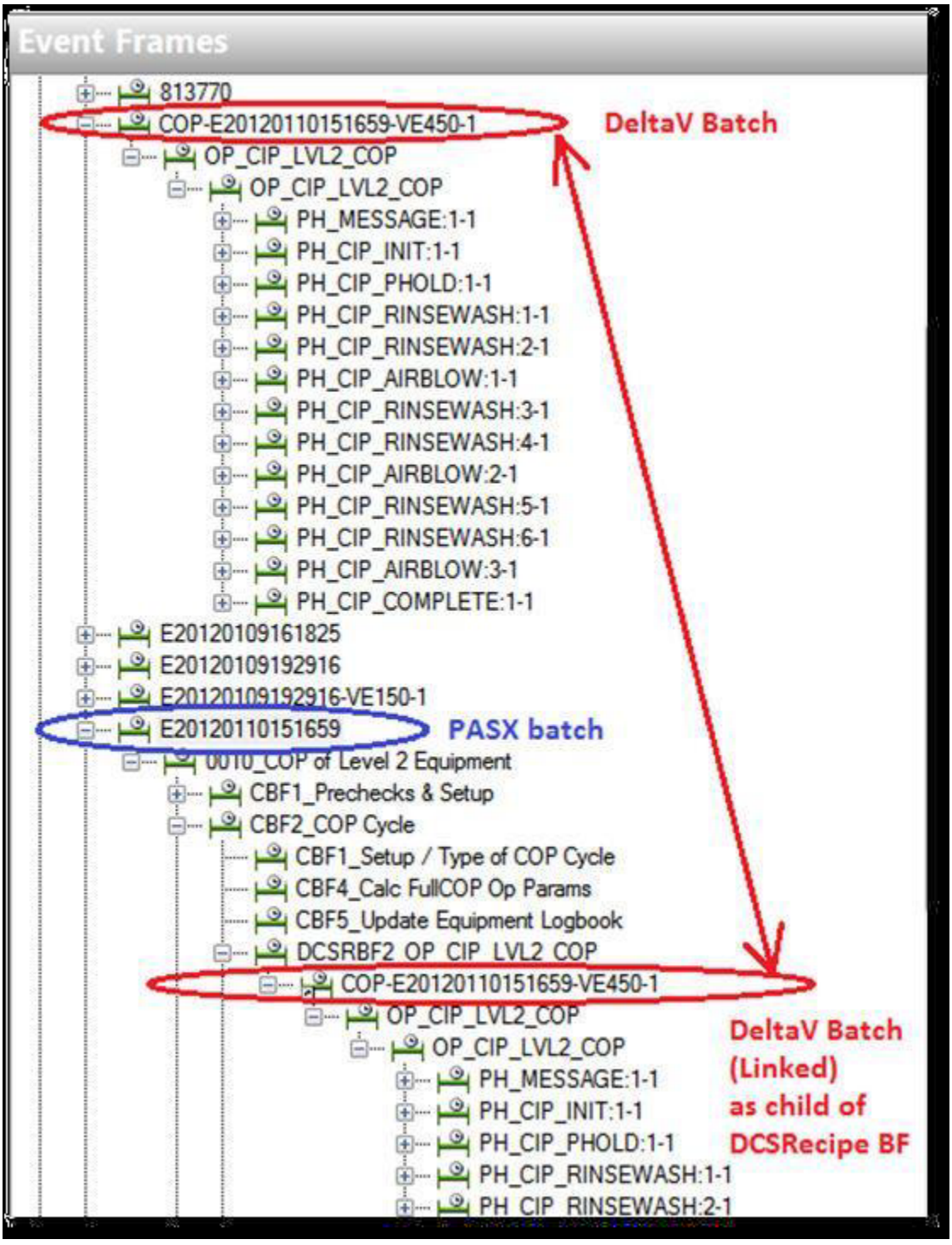
To consolidate the output of a manufacturing execution system (MES) and a batch execution system (BES), specify the linking element as follows:

1. Open PI Event Frames Interface Manager.
2. Click the **Batch Setup** tab.
3. For MES interfaces, enable the **Link to child event frames** option, and specify the path to the PI AF element.
4. For BES interfaces, enable the **Link from parent event frames** option, and specify the path to the same PI AF element.

This configuration is typical in an environment where an MES supervises multiple batch execution systems to manufacture a single product. For each BES, configure a dedicated interface instance and specify the same PI AF linking element. When an event frame created by the MES interface references the creation of a batch in the BES, a reference is made to the MES event frame in the linking element. Then, when the BES interface creates its event frame, a reference to it is created in the MES event frame, creating a consolidated view of all the related events from different systems.

For example, the following figure illustrates an event frame created based on a batch from an Emerson DeltaV batch execution system. This event frame is referenced in an event frame created by a Werum PAS-X MES. Other batch interface instances that are configured to use the same linking element create references to their batches under the same link.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.



The .ini file code for creating those event frames looks like this example:

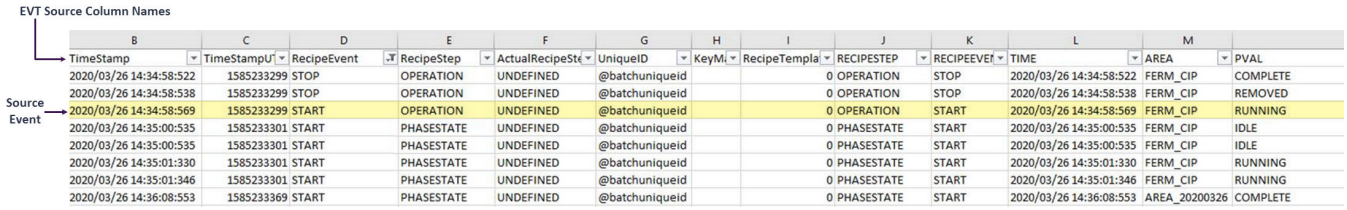
```
HOST=jshearouse5520
PS=EMDVBCS
ID=998
AFHOST=jshearouse5520
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
AFDATABASE=EMDVBCS
DB=2
Source[1].msmqpath=jshearouse5520\private$\historian2
Source[1].websrvpath=example.com
WRITELINK=\\LINKPARENT\LINKTEST
```

Placeholders and advanced parsing

Placeholders enable you to incorporate data from incoming events into tag names and data. Placeholders can be used in all types of templates. The precise set of placeholders supported by an interface depends on the data source. The following example illustrates how placeholders correspond to columns in a data source.



	B	C	D	E	F	G	H	I	J	K	L	M	
	TimeStamp	TimeStampU	RecipeEvent	RecipeStep	ActualRecipeSte	UniqueID	KeyM	RecipeTempla	RECIPESTEP	RECIPEEVEI	TIME	AREA	PVAL
	2020/03/26 14:34:58:522	1585233299	STOP	OPERATION	UNDEFINED	@batchuniqueid		0	OPERATION	STOP	2020/03/26 14:34:58:522	FERM_CIP	COMPLETE
	2020/03/26 14:34:58:538	1585233299	STOP	OPERATION	UNDEFINED	@batchuniqueid		0	OPERATION	STOP	2020/03/26 14:34:58:538	FERM_CIP	REMOVED
	2020/03/26 14:34:58:569	1585233299	START	OPERATION	UNDEFINED	@batchuniqueid		0	OPERATION	START	2020/03/26 14:34:58:569	FERM_CIP	RUNNING
	2020/03/26 14:35:00:535	1585233301	START	PHASESTATE	UNDEFINED	@batchuniqueid		0	PHASESTATE	START	2020/03/26 14:35:00:535	FERM_CIP	IDLE
	2020/03/26 14:35:00:535	1585233301	START	PHASESTATE	UNDEFINED	@batchuniqueid		0	PHASESTATE	START	2020/03/26 14:35:00:535	FERM_CIP	IDLE
	2020/03/26 14:35:01:330	1585233301	START	PHASESTATE	UNDEFINED	@batchuniqueid		0	PHASESTATE	START	2020/03/26 14:35:01:330	FERM_CIP	RUNNING
	2020/03/26 14:35:01:346	1585233301	START	PHASESTATE	UNDEFINED	@batchuniqueid		0	PHASESTATE	START	2020/03/26 14:35:01:346	FERM_CIP	RUNNING
	2020/03/26 14:36:08:553	1585233369	START	PHASESTATE	UNDEFINED	@batchuniqueid		0	PHASESTATE	START	2020/03/26 14:36:08:553	AREA_20200326	COMPLETE

When you define templates using the PI Event Frame Interface Manager, you can choose from a list of supported placeholders. To use a placeholder in a field when editing a template, click the **Add Placeholder** button and choose the desired placeholder.

You can define placeholders that read data from PI tags when triggered by batch events. To specify a tag-based placeholder, use the following syntax:

```
[Tag, Name="PI Tag Name", <comma-delimited list of parameters>]
```

For example:

```
total:[Tag, name="sinusoid", range="10d", func="TOTAL"]

min:[Tag, name="test_data_1", range="10d", func="MIN"]
```

The following table describes the components of a tag-based placeholder.

Parameter	Description
Name="tagname"	(Required) Defines the exact name of the PI tag which should be used for data retrieval.
Range="data_range"	(Optional) Defines the time frame for which the data is queried. It can be number of events, time frame or "PIOBJECT". "PIOBJECT" instructs the interface to use the time frame of the related PI batch/unitbatch/ subbatch object. Examples: Range="10": Retrieve the ten events that precede the triggered batch event timestamp. Range="10d": Retrieve the events for ten days from the triggered batch event timestamp. Range="PIOBJECT": Retrieve the events between the start and end times of the related batch object.
Func="option"	(Optional) Used with Range to aggregate retrieved data. Options:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Parameter	Description
	• "MIN": Minimum value in the time frame. • "MAX": Maximum value in the time frame. • "TOTAL": Sum of values in the time frame. • "MID": Average of values in the time frame.

Following are wildcards, which you can use to define a mask for template settings and placeholders, to match incoming data and format data to be written to the PI System.

Wildcard	Description
#	Single digit numerical value, 0-9
@	Single alpha character, a-z, A-Z
?	Any single symbol
!	Repeat the previous mask symbol
*	Any set of symbols

For example, to match any event that starts with "Repo", use the * wildcard as follows: [EVENT, VALUE="Repo*"].

The interface supports a set of parameters that provide fine control over how incoming data is parsed. These advanced parsing parameters can be used in all types of templates. To enable you to incorporate these parsing parameters into a placeholder expression, the **Build a Placeholder** dialog provides an **Add Substring Parsing** option.

If you specify parsing parameters for one or more placeholders and no matching data is found, the entry is set to blank. The following table lists the parameters for parsing incoming data. The names of parameters, placeholders, and value substrings are not case-sensitive.

Parameter	Description
LBE=”substring”	Defines the left bound of the target substring value. The resulting substring does not include the specified substring.
LBI=”substring”	Defines the left bound of the target substring value. The resulting substring includes the specified substring.
RBE=”substring”	Defines the right bound of the target substring value. The resulting substring does not include the specified substring.
RBI=”substring”	Defines the right bound of the target substring value. The resulting substring includes the specified substring.
DELIM=”substring”	Specifies the field separator character or substring. Must be used in conjunction with the COUNT parameter. It narrows the results to the substring contained in delimiters, where the starting delimiter index is specified by the COUNT parameter. To parse the delimited substring, you can specify right and left boundary substrings.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Parameter	Description
COUNT=#	Index (position) of the delimiter from which to start parsing. Must be used in conjunction with the DELIM parameter.

To search all fields of an incoming event, specify a wildcard for the placeholder name (for example [*,lbe="u:"]).

The following table lists examples of parsing a field that contains the following data:

`|U:browntod|C:SP_CHARGE_AMOUNT|O:1200|N:1123|E:kg|M:Local`

Example	Resulting Data
[value, lbe="N:"]	1123|E:kg|M:Local
[value, lbi="N:"]	N:1123|E:kg|M:Local
[value, rbe="tod"]	|U:brown
[value, rbi="tod"]	|U:browntod
[value, lbe="U:", rbe="|"]	Browntod
[value, lbi="U:", rbe="|"]	U:browntod
[value, lbe="O:", rbi="kg"]	1200|N:1123|E:kg
[value, delim="|",count=3]	O:1200
[value, delim="|",count=3,lbe="O:"]	1200
[value, delim="|",count=3,lbe="C:SP",rbe="UNT"]	_CHARGE_AMO
[value, delim="|",count=6,lbe="M:"]	Local

PIEVENT placeholders

These placeholders are auto generated by the batch interface and triggered by PI batch events (Parameter, value="PIEVENT") or (Event, value="PIEVENT").

Place Holder	Batch Level	How Batch Database is Stored	How Event Frames are Stored
[BATCHID]	1 or 2	String value that is stored as PIBatch BatchID and PIUnitBatch BatchID property	Top-level event frame: Name property Second-level event frame: BatchID attribute.
[PROCEDURE]	1	PIBatch Recipe property	Top level event frame Recipe Attribute
[UNITPROCEDURE]	2	PIUnitBatch Procedure property	Name property
[OPERATION]	3	PISubBatch Name property	Name property
[PHASE]	4	PISubBatch Name property	Name property
[PHASESTATE]	5	PISubBatch Name property	Name property

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Place Holder	Batch Level	How Batch Database is Stored	How Event Frames are Stored
[PHASESTEP]	6	PISubBatch Name property	Name property
[UNIT]	n/a	PIModule Name property	PI AF element Name property.

For example, to update a PI tag when a PI batch is started or ended, specify the following tag templates:

```
Tag[1].Name=BESName:PIEvent.Batch.Active
Tag[1].Value=BATCH START: [BATCHID] |Prod: [PRODUCT] |Rec: [PROCEDURE]
Tag[1].Trigger=[EVENT,value="PIEVENT"] [DESCRIPT, value="BATCH"] [PVAL,value="START"]

Tag[2].Name=BESName:PIEvent.Batch.Active
Tag[2].Value=BATCH END: [BATCHID] |Prod: [PRODUCT] |Rec: [PROCEDURE]
Tag[2].Trigger=[EVENT,value="PIEVENT"] [DESCRIPT, value="BATCH"] [PVAL,value="END"]
```

Templates for mapping data source events

Templates enable you to capture data from the data source, specifying the format for the desired data and the events that cause the interface to capture the data. To configure templates, go to the PI Event Frame Interface Manager Templates tab. You can define the following types of templates:

Template Type	Description
Tag	Create and update PI tags using data from data source.
Property / Attribute	Create and update event frame attributes using data from data source.
Recipe	Override the data source recipe level names.
Alarm tag	Create and update PI tags using data from an Emerson DeltaV alarms and events data source.
Asset attribute	Create and update element attributes using data from the data source.
DCS Template	Identify the value from event frame attributes that contain the batch ID of the batch that we want to link to as a child object.

You can configure templates that map the source data to **string**, **integer** or **float** data types and configure the format of the information to be written to the target. The precise format of the events coming from the data source depends on the BES vendor. For detailed information, refer to the vendor-specific topic.

Tag templates

This section details the procedure for configuring tag templates. The tables in steps 7 and 8 define specific tag template settings and configurations to ensure that tag templates capture updates to PI Batch Database.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

1. To create or update PI tags when specified events are read, configure tag templates.
2. To create or update PI tags based on alarms read from an Emerson DeltaV Event Chronicle (alarms & events) data source, configure alarm tag templates.
3. To define tag templates using PI Event Frame Interface Manager, go to the **Templates** page and click the **Tag** tab.
4. To configure the name of the tag to be created or updated, specify the **Name** field. To assign tag names based on incoming data, use placeholders.

   For example, to track phase module report events on a per-unit basis, you might configure the name as follows:

   ```
   [Unit] [phasemodule] Report
   ```

   With the preceding template, when the interface reads a report event for the NORTON phase module on unit XUNIT_52003, it replaces the placeholders with data from the specified fields and creates or updates a PI tag with the following name:

   ```
   XUNIT_52003 NORTON Report
   ```

   If the name structure contains placeholders, the tag template is triggered only if all the corresponding fields from the incoming event contain data (that is, are not blank).

   Different templates can update the same PI tag, if the templates' name structure resolves to the same tag. This capability enables you to write different values to the tag depending on the nature of the triggering event. For example, a value of 1 can be written to the tag when a unit procedure starts and a value of 0 can be written to the same tag when the unit procedure ends.
5. To specify the data to be written to the tag, configure the **Value** field. To include data read from the data source in the tag value, use placeholders.

   For example, to simply record the incoming value without transforming it, specify the [PVAL] placeholder. A more complex example: to configure a value that concatenates phase module, event, description, incoming value and engineering units, specify the following:

   ```
   [PHASEMODULE].[EVENT].[DESCRIPT]: [PVAL] [EU]
   ```

   The preceding expression generates data like the following:

   ```
   CHARGE_DIW.Recipe Value.CPP_HIGH_LIMIT: 2535 kg
   ```

   Unlike placeholders in tag names, value placeholders can be replaced with empty fields from the incoming event, unless you use advanced field parsing to configure the value.
6. To update a tag when a particular event is read from the data source, specify the EVENT keyword in the **Name** field, as follows:

   ```
   [EVENT, VALUE="event_text"]
   ```

   This approach enables you to write different values to the tag depending on the text in the EVENT column.

   If you require a more refined approach, specify the incoming data that causes the template to be evaluated by configuring one or more triggers on the **Trigger** tab of the tag template.

   To configure the template to handle multiple different events, specify separate triggers ("OR" logic). To ensure that the template is triggered only when a set of multiple conditions are all detected ("AND" logic), specify a single trigger containing all the conditions. For example, to trigger the template only for system message events that are phase logic failures, specify the trigger as follows:

   ```
   [EVENT, value="System Message"] [DESCRIPT, value="Phase Logic Failure"]
   ```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

To ignore specified incoming values, use "!=" (not equal) . For example, to ignore undefined values, specify the following expression:

```
[PVAL, VALUE!="UNDEFINED"]
```

You can use wildcards to specify pattern-matching expressions in triggers.

7. To configure the tag template settings, specify settings as described in the following table:

Setting	Description
NAME	(Required) Name of PI tag to be created or updated.
VALUE	(Required) Value to be assigned to PI tag (text)
TRIGGER	Event text from data source (can be specified using wildcards)
TYPE	String/integer/float/auto. "Auto" directs the interface to automatically detect the data type.
UNITALIAS	Configure how unit alias (AF: PI point reference) is created. By default, the alias is created in the unit. To override the default, specify the path where you want the alias created. For example: `UNITALIAS = \Building1\Unit2
PHASEALIAS	Configure how the phase alias is created. By default, the alias is created in the phase module. To override the default, specify the path where you want the alias created.
DESCRIPTOR	Value for PI point descriptor attribute.
ENGUNITS	Engineering units
TRANSLATE	To enable translation, set to TRUE (default: FALSE)
ANNOTATION	Simple annotation to be written to the tag when the interface updates it.
ANNOTATION2	Structured annotation to be written to the tag when the interface updates it. For details about structured annotations, refer to the PI Data Archive System Management Guide.

8. To configure tag templates that catch events raised by the interface when it updates the PI Batch Database, specify the following placeholders in the TRIGGER setting of the tag template:

Placeholder	Values	Description
EVENT	**PIEVENT**	Specify [EVENT, value="PIEVENT"]
DESCRIPT	**BATCH** **UNITBATCH** **OPERATION** **PHASE** **PHASESTATE** **PHASESTEP**	Specify the batch level you want to trigger on. For example: [DESCRIPT, value="UNITBATCH"] [DESCRIPT, value="PHASE"]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Placeholder	Values	Description
PVAL	**START** **END**	Specifies whether to catch the start or ending event of the specified level: [PVAL, value="START"] [PVAL, value="END"]

For example, to detect the start of a batch, specify the following expression:

```
[EVENT, VALUE="PIEVENT"][DESCRIPT, VALUE="BATCH"][PVAL, VALUE="START"]
```

The following placeholders are supported when the triggering expression contains `[Parameter, value="PIEVENT"]`.

Placeholder	Batch Database	Event Frames
[BATCHID]	PIBatch and PIUnitBatch: BatchID property.	For a top-level event frame, "Name" property. For second-level event frame, "BatchID" attribute
[PROCEDURE]	PIBatch "Recipe" property	Event frame "Recipe" attribute
[UNITPROCEDURE]	PIUnitBatch "Procedure" property	Event frame "Name" property
[OPERATION]	PISubBatch "Name" property	Event frame "Name" property
[PHASE]	Level 4 PISubBatch "Name" property	Event frame "Name" property
[PHASESTATE]	Level 5 PISubBatch "Name" property	Event frame "Name" property
[PHASESTEP]	Level 6 PISubBatch "Name" property	Event frame "Name" property
[UNIT]	PIUnit "Name" property	Event frame "Name" property
[PHASEMODULE]	Phase module "Name" property	Event frame "Name" property

Tag template examples

Capture report events for phase modules

Tag template

```
Tag[1].Name =[Unit] [phasemodule] [Event, value="Report"]
Tag[1].Value =[Descript]:[Pval]:[EU]-[Event]_Just for testing
Tag[1].Type = string
Tag[1].unitalias=[phasemodule] Report
Tag[1].phasealias=Report
Tag[1].Annotation=[BatchID]
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Incoming data

```
[BatchID]=Batch_123
[Time]=08/12/2008 13:00:05.123
[Descript]= Level
[Pval]=123.456
[EU]=cm
[Event]=Report
[Unit]=U101
[PhaseModule]=PDL123
```

Resulting PI point: `U101 PDL123 Report`

Resulting event added to PI point:

- Timestamp: 08/12/2008 13:00:05.123
- Value: Level:123.456:cm-Report_Just for Testing
- Annotation: Batch_123

The MDB module or PI AF asset for this phase module will have an alias that refers to this point.

Multiple triggers for same tag

Tag template

```
Tag[1].Name = [Unit] Tester
Tag[1].Value =[pval]-[Event]
Tag[1].Type = string
Tag[1].Trigger = [Event, value="Report"]
Tag[1].Trigger = [Event, value="State Change"]
Tag[1].Trigger = Comment
Tag[1].unitalias = Testing [Unit]
Tag[1].phasealias = Testing phase alias
```

Incoming data

```
Event 1:
[Time]=08/12/2008 13:00:05.123
[Event]=Report
[Unit]=U101
[Pval]=123.456

Event 2:
[Time]=08/12/2008 15:20:01.998
[Event]=State Change
[Unit]=U101
[Pval]=HOLDING
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
Event 3:
[Time]=08/12/2008 18:49:12.020
[Event]=Comment
[Unit]=U101
[Pval]=Discard Product
```

Resulting PI point: `U101 Tester`

Resulting event added to PI point

TimeStamp	Value (string)
08/12/2008 13:00:05.123	123.456-Report
08/12/2008 15:20:01.998	HOLDING-State Change
08/12/2008 18:49:12.020	Discard Product-Comment

Recording start and stop events

The following tag templates store 1 when an operation or phase starts and 0 when it ends.

```
Tag[1].Name = Activity on [Unit] [Pval]
Tag[1].Value = 1
Tag[1].Type = integer
Tag[1].Trigger = [Descript, value="Step Activated"] [Event, value="Step Ac
tivity] [EU, value="Operation"]
Tag[1].Trigger = [Descript, value="Step Activated"] [Event, value="Step Ac
tivity] [EU, value="Phase"]

Tag[2].Name = Activity on [Unit] [Pval]
Tag[2].Value = 0 Tag[2].Type = integer
Tag[2].Trigger = [Descript, value="Step Deactivated"] [Event, value="Step
Activity] [EU, value="Operation"]
Tag[2].Trigger = [Descript, value="Step Deactivated"] [Event, value="Step
Activity] [EU, value="Phase"]
```

Tag aliasing

The following tag templates create tags corresponding to the aliases shown in the figure below.

```
Tag[1].Name = Super Test1 [Unit]
Tag[1].Value = [Event] | [Pval]
Tag[1].Trigger = [Event,value=State Change]

Tag[2].Name = Super Test2 [Unit]
Tag[2].Value = [Event] | [Pval]
Tag[2].Trigger = [Event,value=State Change]
Tag[2].UnitAlias = stuff2\[Event] | Super Testing2

Tag[3].Name = Super Test3 [Unit] [PhaseModule]
```

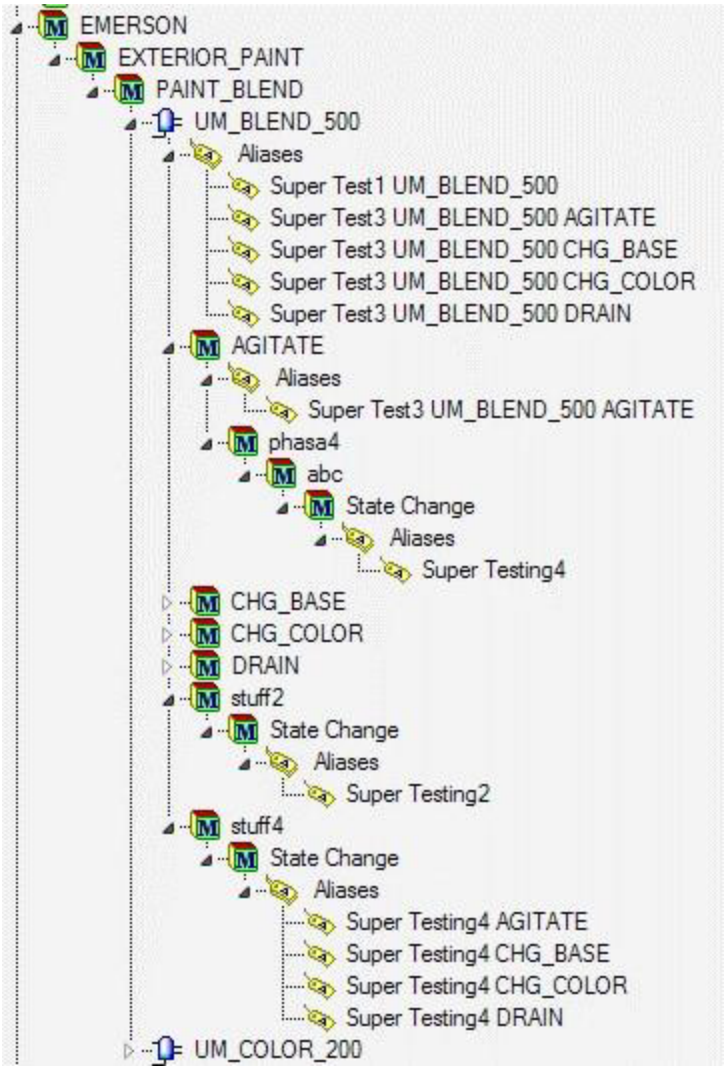
©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
Tag[3].Value = [Event] | [Pval]
Tag[3].Trigger = [Event,value=State Change]

Tag[4].Name = Super Test4 [Unit] [PhaseModule]
Tag[4].Value = [Event] | [Pval]
Tag[4].Trigger = [Event,value=State Change]
Tag[4].UnitAlias = stuff4\[Event] | Super Testing4 [PhaseModule]
Tag[4].PhaseAlias = phasa4\abc\[Event] | Super Testing4

Tag[5].Name = Super Test5 [Unit] [PhaseModule]
Tag[5].Value = [Event] | [Pval]
Tag[5].Trigger = [Event,value=State Change] Tag[5].UnitAlias = NONE
Tag[5].PhaseAlias = NONE
```

Tag template example



©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Link templates

DCS Link Templates associate event frames from one interface to another. To create links, you define link templates on the **Templates** page of the PI Event Frame Interface Manager. The link templates specify events that contain link data and the naming of the destination event frame. Placeholders, wildcards and advanced parsing features enable you to extract precisely the desired data from the events that the interface reads from the data source.

Triggers on the template define the events with linking information and the value specifies the name of the destination batch level event frame. To define a link template on the **Templates** page of the PI Event Frame Interface Manager, configure the following settings:

Setting	Description
Value	The name of the destination batch level event frame.
Trigger	The event that triggers the generation of the property. Updates can be triggered by the data source or by PI events such as the start or end of a recipe level.

Examples

To link an event frame that has an event where `Descript` is `WorkflowID` to a batch level event frame whose name is held in `Pval` in the same event, configure the template with the following settings:

Setting	Set to
Value	[Pval]
Trigger	[Descript, value="WorkflowID"]

Property templates

Properties associate data from the data source with a desired level of the batch or event frame hierarchy. To create properties, you define property templates on the **Templates** page of the PI Event Frame Interface Manager. The property templates specify the data to be extracted and the level of the hierarchy at which they are created. Placeholders, wildcards and advanced parsing features enable you to extract precisely the desired data from the events that the interface reads from the data source.

With event frames, properties (AF attributes) can be created at any level of the batch hierarchy. However, in the batch database, properties are stored only at the top (procedure or batch) level of the batch hierarchy. This top-level collection of properties includes the properties for the lower levels of the batch hierarchy, arranged according to level.

To define a property template on the **Templates** page of the PI Event Frame Interface Manager, configure the following settings:

Setting	Description
Index	Unique numeric ID (integer).
Name	Property name. Every event name under the same node must be unique. If the template does not define a name, the interface assigns a default name of

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Setting	Description
	`Event_<n>`, where `<n>` is a unique integer. To write properties at the top level regardless of the level from which the triggering event originated, specify the `@` symbol as the first element in name path as follows: `@\[Parameter]`. To write properties under the UniqueID property (PI Batch only), regardless of the level from which the triggering event originated, specify the `$` symbol as the first element in name path, as follows: `$\[Parameter]`.
Value	Value to assign to property.
Create property	Specifies the level at which the property is to be created. Options are as follows: • At recipe level: By default, properties are created at the recipe level corresponding to the level in the data source from which the data was read. • Under root node: Create properties at the top level of the batch or event frame hierarchy, regardless of the level at which they originated. • Under UniqueID node: (Batch database only, not event frames) Create properties under the batch's UniqueID node.
Data type	PI data type of the value. Note that if the incoming value is incompatible with the specified type, the event is not processed, and an error is logged.
Trigger	The event that triggers the generation of the property. Updates can be triggered by the data source or by PI events such as the start or end of a recipe level.
Translate	Specifies whether the property is to be translated before being written to the PI System. Specify true or false (default is false).
EngUnits (AF only)	Engineering units
Type	`INTEGER`, `FLOAT`, `STRING` (Default), `FLOATSTRING`, `AUTO`. If Type is set to `AUTO`, the interface assigns the data type based on the first item of data that it receives. If the first value received is an integer but subsequent values are floating point, the interface creates an integer attribute or property and truncates any subsequent floating point values before storing them. To avoid truncating numeric input that includes a mix of integer and floating point values, specify `FLOATSTRING` rather than `AUTO`.
Category (AF only)	Asset category

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Setting	Description
UOM (AF only)	Unit of measure. To map the units of measure in the data source to the correct PI AF engineering units, check the UOM check box and configure the mapping between the source units of measure and the units available in PI AF.
Descriptor (AF only)	Description.
Tagpath	The path to a PI Server tag. For example: `\\MyPIServer\sinusoid`. This will create an attribute that has a data reference to a PI point.
Allowemptyvalue	If true, the attribute will be created with an empty value. If false, and the value is empty, the attribute will not be created.

Examples

To create or update a property named TestTagCalc that contains a ten-day total for the "sinusoid" tag and a ten-day minimum for the "test_data_1" tag, configure the template with the following settings:

Setting	Set to
Name	TestTagCalc
Value	total:[Tag, name="sinusoid", range="10d", func="TOTAL"]
Trigger	[Event,value="PIEVENT"] [Descript,value="BATCH"] [Pval,value="START"]

To populate the product property with the formula name when a "Recipe header" event arrives, ignoring undefined product codes, use the following settings:

Setting	Set to
Product	[PVAL]
ProductTrigger	[EVENT,VALUE="Recipe Header"] [DESCRIPT,VALUE="Product Code"] [PVAL, VALUE! ="UNDEFINED"]
ProductTrigger	[EVENT,VALUE="Formula Header"] [DESCRIPT,VALUE="Formula Name"]

Recipe templates

The interface includes a set of built-in default recipe templates that control the name assigned to each level in the batch hierarchy and the data stored at each level. To override the naming convention and data assigned to PI batches, unit batches, subbatches and event frames, you define recipe templates. To define recipe templates, launch the Event Frames Interface Manager and perform the following steps.

1. Go to the **Templates** page and navigate to the list of recipe templates.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

2. Right-click the top Recipe node and choose **Add Default Templates**.

   A list of default recipe templates displays.

3. For the recipe levels that you want to modify, configure basic settings as follows:

Setting	Description
NAME	(Required) Defines the convention used by the interface to assign names to procedures, unit procedures, etc. You can use the advanced parsing parameters to define this field. Example: `abc_[PROCEDURE]` If the procedure field of the incoming event contains `Test`, the resulting name is `abc_Test`.
BATCHID	(Optional) Specifies the batch ID for the procedure or unit procedure.
MODULEPATH	(Optional) For unit procedures (level 2) or phase (level 4), specifies where the recipe resides in the PI Module Database or PI AF element hierarchy. In the Module Database, the path specifies the location of the PIUnit for the unit batch.
PRODUCT	(Optional) Set the product for the recipe. Sets the Product field for procedures and unit procedures. To set the product field to the value read from the data source, specify the following placeholder: `[PRODUCT]`
PRODUCTTRIGGER	(Optional) Sets the product for the recipe after the recipe object is created. Intended for use when the product is defined in a separate event. If a product trigger is defined, the product is defined by the event that satisfies the trigger. If no product trigger is defined, the product gets its value from the event that created the recipe, and the template is populated by the event's placeholder data. Example: `[Parameter, Value="Recipe Header"] [Descript, value="Product Name"]`
TRANSLATE	(Optional) To enable translation, set to `TRUE`. Default: `FALSE`
MERGE	(Optional) To merge identically named objects under the same parent, set to `TRUE`. Default: `FALSE`

   For more information on the placeholders supported for each setting, see Template placeholders

4. For event frames, you can configure the following settings.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Setting	Description
Descriptor	(Optional) Specifies the Event frame descriptor property for the particular source Recipe object.
DefaultProperty[x].Name	(Optional) Name of the event frame template attribute. Valid values are Recipe, BatchID, Product and Procedure. Interface defined defaults place Recipe and BatchID at x=1, Product at x=2, and Procedure at x=3.
DefaultProperty[x].Value	(Optional) Defines the event attribute expression that evaluates to a valid value.
DefaultProperty[x].Trigger	(Optional) Defines the expression that specifies which event(s) to use to get the value. Example: `[Event,value="Recipe Header"][descript,value="Product Code"]`
DefaultProperty[x].UseFirstValue	(Optional) Use the first matching event for the event frame to get the value if set to T or True. The default behavior is to use the last matching event.
Category	(Optional) For each recipe level, defines the event frame category. If the event that creates an event frame contains insufficient information, no category is assigned. To assign a category to an event frame after its creation, use Category[x].
Category[x].Name	(Optional) For each recipe level, define the event frame category based on an event that is related to the particular recipe item. This setting can create as many categories as desired. The index is a positive integer that associate the Name and Trigger subproperties for the specific Category[x] property. If the AF category does not exist, the interface creates it. To use this setting, you must also specify the triggering event using the Recipe[#].Category[x].Trigger setting. Example: When the specified trigger event arrives, create an event frame, assigning it the SCR category. `Category[10].Name = SCRCategory[10].Trigger = [Descript, value="Formula Name"][Pval, value="SCR 20051"]`
Category[x].Trigger	(Optional) Defines the expression that triggers assignment of a category by the Recipe[#].Category[x] setting. There can be multiple

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Setting	Description

	triggers for a single Recipe[#].Category[x].Name. To use this setting, you must also specify the category to be assigned, using the Recipe[#].Category[x].Name setting.
	
	Example: When any of the specified trigger event arrives, create an event frame, assigning it the SCR category.
	
	`Category[10].Name = SCR`
	
	`Category[10].Trigger = [Descript, value="Formula Name"] [Pval, value="SCR 20051"]`
	
	`Category[10].Trigger = [Descript, value="Formula Name"] [Pval, value="SCR 20051_01"]`
	
	`Category[10].Trigger = [Descript, value="Formula Name"] [Pval, value="SCR 20051_02"]`

Template	(Optional) For each recipe level, specify the event frame template. If the interface cannot find a matching event frame template, the template is left blank. To assign a template to an event frame after its creation, use the Template[x] property.

Recipe[#].Template[x].Name	(Optional) For each recipe level, this dynamic property enables you to define the event frame template. Based on any event that is related to particular recipe item. This property can assign only one AF template to a particular event frame. The interface uses the first matching Recipe[#].Template[x] property to be assigned to an event frame. The index is a positive integer that associates the Name and Trigger subproperties for the Template[x] property. If you specify this property, you must specify the Trigger property.
	
	Example:
	
	`Template[10].Name = BATCH_A`
	
	`Template[10].Trigger = [Descript,`

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Setting	Description
	`value="Formula Name"] [Pval, value="SCR 20051"]`

Recipe[#].Template[x].Trigger	(Optional) This property defines the triggering expression for the event frame template. There can be multiple triggers for a single recipe template. If you specify this property, you must specify the Name property.
	
	Example:
	
	`Template[10].Name = BATCH_A`
	
	`Template[10].Trigger = [Descript, value="Formula Name"] [Pval, value="SCR 20051"]`
	
	`Template[10].Trigger = [Descript, value="Formula Name"] [Pval, value="SCR 20051_01"]`
	
	`Template[10].Trigger = [Descript, value="Formula Name"] [Pval, value="SCR 20051_02"]`

5. To save your changes, click **Save Settings** in the left pane.

Recipe template examples

Configure the batch name as the concatenation of the procedure and unique ID fields from the data source, and name operations by concatenating the parent unit procedure name with the operation name:

```
Recipe[1].Name = [Procedure]_[UniqueID]
Recipe[3].Name = [UnitProcedure]_[Operation]
```

Assign the PI AF category "CAT_UNITBATCH" to all unit procedures. Note that, if the specified category has not been defined in PI AF, the interface creates it.

```
Recipe[2].Category = CAT_UNITBATCH
```

Dynamically assign categories based incoming data:

```
Recipe[1].Category[1].Name = PROC_A
Recipe[1].Category[1].Trigger = [Procedure, value="DVProc:1-*"]
Recipe[1].Category[2].Name = [Pval]
Recipe[1].Category[2].Trigger = [Descript, value="Product Code"]
```

When creating event frames for unit procedures (level 2), use the corresponding template:

```
Recipe[2].Template = OSI_UnitProcedure
```

Dynamically assign templates based incoming data:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
Recipe Template Example 5:
Recipe[2].Template[1].Name = UP_A
Recipe[2].Template[1].Trigger = [UnitProcedure, value="UProc:1-*"]
Recipe[2].Template[2].Name = UP_B
Recipe[2].Template[2].Trigger = [UnitProcedure, value="UProc:2-*"]
```

Recipe templates default properties

Batch Level

The following are the recipe template default values for the BATCH level:

Template	Default
Name	[Procedure]
Value	[BatchID]
Product	[pval]
Product Trigger	[Event,value="Recipe Header"] [descript,value="Product Code"])
Category	OSIBatch
Template	Procedure

Default Properties

[RECIPE]

The index in the recipe template heirarchy corresponds to the RECIPE item default property (1).

```
  Recipe[1].DEFAULTPROPERTY[1].NAME = Recipe
  Recipe[1].DEFAULTPROPERTY[1].VALUE = [PROCEDURE|UNITPROCEDURE|OPERATION|
Phase|PhaseState|PhaseStep]
```

[PRODUCT]

The index in the recipe template heirarchy corresponds to the PRODUCT item default property (2).

```
  Product[2].DEFAULTPROPERTY[2].NAME = Product
  Product[2].DEFAULTPROPERTY[2].VALUE = [pval]
  Product[2].DEFAULTPROPERTY[2].TRIGGER = [Event,value="Recipe Header"][de
script,value="Product Code"]
```

Unit Batch Level

The following are the recipe template default values for the UNIT BATCH level:

Template	Default
Name	[UnitProcedure]
Value	[BatchID]
Product	[pval]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Template	Default
Product Trigger	[Event,value="Recipe Header"] [descript,value="Product Code"])
SearchByStartTime	true
SearchByEndTime	true
Category	OSIBatch
Template	UnitProcedure
Module Path	[AREA]\[PROCESSCELL]\[UNIT])

Note - Unitbatches can have the same name when SearchByStartTime and SearchByEndTime equal true.

Default Properties

[PROCEDURE] = 3

```
   .NAME = PROCEDURE
   .VALUE = [UNITPROCEDURE|OPERATION|Phase|PhaseState|PhaseStep]
```

[PRODUCT] = 2

```
   .NAME = Product
   .VALUE = [pval]
   .TRIGGER = [Event,value=\"Recipe Header\"][descript,value=\"Product Code
\"]
```

[RECIPE] = 1

```
   .NAME = BatchID
   .VALUE = BatchID
```

Operation Level

The following are the recipe template default values for the OPERATION level:

Template	Default
Name	[Operation]
Category	OSIBatch
Template	UnitProcedure

Phase Level

The following are the recipe template default values for the PHASE level:

Template	Default
Name	[Phase]
Category	OSIBatch
Template	Phase
Module Path	[AREA]\[PROCESSCELL]\[UNIT]\[PHASEMODULE])

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Phase State Level

The following are the recipe template default values for the PHASE STATE level:

Template	Default
Name	[PhaseState]
SearchDirection	Reverse
SearchByStartTime	true
SearchByEndTime	true
Category	OSIBatch
Template	PhaseState

Phase Step Level

The following are the recipe template default values for the PHASE STEP level:

Template	Default
Name	[PhaseStep]
Merge	false
SearchDirection	Reverse // search in Reverse (default: always FORWARD)
Category	OSIBatch
Template	PhaseState

Filter incoming events

You can configure filters to restrict the interface from processing certain events. You can filter out data based on its level in the batch hierarchy and on its unit. The interface examines fields in incoming events for the specified data and, if it matches, the event is not processed.

1. To exclude incoming data from processing using PI Event Frames Interface Manager, go to the Filters tab.
2. Expand the setting for the type of data that you want to filter (phases, units, recipes or phase states).
3. Specify the data that you want to exclude.

   You can use wildcards to define masks for matching.

Translate foreign language data

To process data from non-native language data sources, you can translate strings from the source language to the target language.

1. To configure a translation using PI Event Frame Interface Manager, open the desired template.
2. Select the **Translate** check box and click **Translations**.

   The Translations window opens.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

3. Specify the desired source and target text.
   Translations are not case-sensitive.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

PI Interface for Emerson DeltaV Batch

PI Interface for Emerson DeltaV Batch supports all process components and is designed to be used for DeltaV 10.3 and later systems utilizing the DeltaV OPC A&E Server and the DeltaV Batch Historian. Additionally, you can run the interface against DeltaV 9.3 and DeltaV 8.4 by utilizing different data sources.

A single DeltaV interface instance can collect data from multiple data sources provided all sources are of the same type. This includes DeltaV Batch Historian, Event journal files, DeltaV Alarms and Events server and OPC Alarms and Events server. This interface automatically creates Batches in the PI Batch Database or Event Frames in PI Asset Framework (AF) with corresponding representational asset hierarchies in the Module Database (MDB) or PI AF. Process data can be stored in automatically created PI Tags or in properties attached to Event Frames/ Batches based on a flexible template methodology.

Data sources

PI Interface for Emerson DeltaV interface can collect data from the following types of data sources:

- DeltaV Batch Historian (SQL Server)
- Event journal files (EVT files)
- DeltaV Alarms and Events server (alarms and events data only, not batch events)
- OPC Alarms and Events servers (real-time batch data only, not alarms and events)

A single interface instance can collect data from multiple data sources if the data sources are all the same type.

For DeltaV 9.3 systems, this interface can use the DeltaV Batch Historian or DeltaV event files as the primary data source. For DeltaV 8.4 systems, this interface can only use DeltaV event files as the primary data source.

Note: The use of DeltaV event files as a public interface for the DeltaV System is not recommended by Emerson.

The interface can read data from batches that have been archived in DeltaV, only if it runs under an account that has read access to the DeltaV archive databases.

To generate equipment assets in PI that correspond to the equipment in a DeltaV Alarms and Events server, export the EquiHeir.xml from DeltaV and copy it to a location that the interface can access. On the **Source** page of the PI Event Frame Interface Manager, configure the location of the file in the DeltaV equipment hierarchy file field.

Note: For details about migrating from the Batch Event File Interface to the Emerson DeltaV batch interface, refer to the OSIsoft Knowledge Base article, Migration from Batch Event File (EVT) Interface to Batch Interface Framework, DeltaV Batch Interface, or RtReports.

DeltaV Batch Historian (SQL Server)

The default name of the database where the DeltaV Batch Historian records events is **DVHisDB**. For details about the views from which the interface reads the start and end events for the batch, refer to Start and stop events.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

To communicate with SQL Server, the interface requires the Microsoft ADO driver for Microsoft SQL to be installed on the interface node. The driver is part of SQL Native client package and can be downloaded from the MSDN web site.

Event Journals

Event journals are text files in which the BES logs batch events. For the Emerson DeltaV systems, the interface expects each line in the event journal to be composed of the following tab-delimited fields, in the order specified:

```
[TIMESTAMP] [BATCHID] [RECIPE] [DESCRIPT] [EVENT] [PVALUE] [EU] [AREA]
[PROCCELL] [UNIT] [PHASE] [PHASEDESC] [USERID] [UNIQUEID] [COMMENT]
```

Note: In event journals, the product ID information is stored in the [PVALUE] field, in the row that contains the **Product Code** description. Typically, this is a recipe header event. If the BES indicates the product ID using a value other than Product Code, you must translate the value to Product Code to ensure that the interface detects it. To configure the translation using PI Event Frame Interface Manager, check the Translation field on the template's Configuration tab and specify the source and target string on the Translations dialog.

Emerson DeltaV Alarms & Events

The interface can be configured to read alarms and events data (not batch data, however) from an Emerson DeltaV Event Chronicle data source (version 10.3 or higher).

To read from an Emerson DeltaV Event Chronicle data source, you must install the Microsoft SQL Native Client on the interface node. Delta V Event Chronicle stores its data using Microsoft SQL Server. The default alarms and events database server is named "DELTAV_CHRONICLE" and the default database is named "EJournal". The interface reads events from the Journal table.

To configure an alarms and events data source using PI Event Frames Interface Manager, perform the following steps:

1. On the **Data Source** tab, right-click the **Sources** node and choose **Add SQL Source**. The Configuration page is displayed.
2. Check **Use Alarm and Events Historian**.
3. To specify mappings that ensure that process cell data is recorded correctly (DeltaV does not emit process cell information), click **Area to Process Cell...** or **Unit to Process Cell...** and specify the process cell to be recorded with its events.
4. To configure the interface to use the equipment hierarchy XML file that is generated by DeltaV, specify the path to the XML file in the **DeltaV equipment hierarchy** field.

OPC Alarms and Events

The OPC Alarms and Events (OPCAE) server provides real-time batch data (not alarms and events data) to the batch interface. The interface's recovery feature cannot be used to recover data. When collecting data from an OPCAE server, the batch interface collects only start and end times and equipment allocation data.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

To configure an OPCAE server as a data source, click the **DeltaV OPCAE disclosure** and specify the host node and server name. When you configure an OPCAE server, the DeltaV Batch Historian serves as a backup source and the source for additional batch-associated events.

The interface creates a batch when it identifies the recipe type of the batch, which normally occurs when the recipe is loaded and started. The interface starts a batch when it reads the BATCH-EVENT event with its event attribute [6] = "REMOVED". The recipe structure is read from the events that trigger the start and end of levels, such as Procedure Started/Finished, UnitProcedure Started/Finished, etc.

For all levels below batch, the interface checks event attribute [6] for the value "State Change" and parses event attribute [8] to determine whether the event was a start or end event and to determine which level of the batch hierarchy is affected, as shown in the following table:

Event Attribute [6]	Event Attribute [8]	
State Changed	Start	Stop
	<batch recipe hierarchy> RUNNING	<batch recipe hierarchy> COMPLETED or ABORTED or STOPPED

Start and stop events

The start and end times set by the interface depend on the type of data source and whether the "Use Batch Recipe" (UBR) option is enabled. The UBR option is provided for backward compatibility with the interface.

For more information on how the interface determines start and end times, refer to the sections below.

Event Files Data Source

The interface detects start and end events by scanning the [EVENT] field for "State change" events. To determine whether the event is a start or end, the interface scans the [PVALUE] field. To determine the level that started or ended, the interface checks the [RECIPE] field.

EVT File Default Start and Stop Events

Recipe Level	[PVALUE] Start	[PVALUE] End	Remarks
Batch	"CREATED"	"REMOVED"	
Unit batch	"RUNNING"	"COMPLETE", "STOPPED", or "ABORTED"	Requires equipment acquisition or release event: [EVENT] = "Recipe Arbitration" and [DESCRIPT] = "Resource Acquired by recipe" or "Resource Release by recipe". For unit batch start time, the interface

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

Recipe Level	[PVALUE] Start	[PVALUE] End	Remarks
			uses the timestamp of the start event or equipment acquisition event, whichever is latest. For end time, it uses the timestamp of the end event or the equipment release event, whichever is earliest. The interface reads the unit name from the [PVALUE] field.
Operation	"RUNNING"	"COMPLETE", "STOPPED", or "ABORTED"	
Phase	"RUNNING"	"COMPLETE", "STOPPED", or "ABORTED"	
Phase state	"RUNNING"	n/a	The state of a phase state ends any preceding phase state. If the interface detects multiple start or end events for the same phase state, it uses the first event.
Phase step	n/a	n/a	To enable phase steps, specify the /RAS flag in the INI file or, using PI Event Frames Interface Manager, go to the **Batch Setup** page and enable **Report As Step** and specify start and end text. Phase steps do not create higher-level procedures, unit procedures, and so on, if a parent phase is not found. If no phase step close event is found, the phase step is closed when its parent operation is closed. The interface ignores any zero-duration phase steps.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

EVT File "Use batch recipe" (UBR) Enable Start and Stop Events

EVT files are used to enable batch start and stop events.

Batch Recipe Level

Either of the following enable **Batch** level recipe start events:

[EVENT] = "System Message" and [PVALUE] = "Beginning Of BATCH"

or

[EVENT] = "State Change" and [PVALUE] field = "RUNNING"

Either of these events end a **Batch**:

[EVENT] = "System Message" and [PVALUE] = "End Of BATCH"

or

[EVENT] = "State Change" and [PVALUE] field = "REMOVED", "COMPLETE" or "ABORTED" or "STOPPED" or "ABANDONED"

Unit Batch Recipe Level

These events **Start** a **Unit Batch** level recipe:

[EVENT] = "System Message" and [DESCRIPT] field = "Unit Procedure Started"

and

[EVENT] field = "Recipe Arbitration", [DESCRIPT] field = "Resource Acquired by recipe"

and

[EU] field = "Unit". The [PVALUE] field contains the actual unit name.

The latest timestamp is used as the start time for the Unit Batch.

[EVENT] = "State Change" and [PVALUE] field = "RUNNING"

These events **End** a **Unit Batch** recipe:

[EVENT] = "System Message" and [DESCRIPT] field = "Unit Procedure Finished"

and

[EVENT] field = "Recipe Arbitration", [DESCRIPT] field = "Resource Released by recipe"

and

[EU] field = "Unit". The [PVALUE] field contains the actual unit name.

The earliest timestamp is used as the end time.

Unit Batch end if [EVENT] = "State Change" and [PVALUE] is "REMOVED" or "COMPLETE" or "ABORTED" or "STOPPED" or "ABANDONED"

Operation Level Recipe

The following event is the **Start** of an **Operation** level recipe:

[EVENT] = "System Message" and [DESCRIPT] = "Operation Started"

Either of these events End an Operation level recipe:

[EVENT] = "System Message" and [DESCRIPT] field = "Operation Finished"

or

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

[EVENT] = "State Change" and [PVALUE] field = "REMOVED" and [RECIPE] = "Operation"

The earliest timestamp is used.

Phase Level Recipe

The event is the **Start** of a **Phase** level recipe:

[EVENT] = "State Change" and [PVALUE] = "RUNNING"

This event is the **End** of a **Phase** level recipe:

[EVENT] = "State Change" and [PVALUE] = "RUNNING" or "COMPLETE" or "ABORTED"

SQL Batch Historian

The DeltaV BES stores its batch execution history using Microsoft SQL Server. The interface retrieves history from the DVHisDB database.

SQL Default Start and Stop Events

Recipe Level	[EVENTTYPE]	[EVENTDESCRIPT] Start	[EVENTDESCRIPT] End
Batch	"State Change"	CREATED	REMOVED
Unit batch	"State Change"	RUNNING	COMPLETE, STOPPED or ABORTED
Operation	"State Change"	RUNNING	COMPLETE, STOPPED or ABORTED
Phase	"State Change"	RUNNING	COMPLETE, STOPPED or ABORTED
Phase state	n/a	User-specified text	Last phase state in phase: COMPLETE, STOPPED or ABORTED or parent level ends. Preceding phases end when next phase state starts.

To detect start and end events, the interface queries the following views by default:

- brecipestatechangeview
- batcheventview
- batchequipmentview

By default, batch event start and end times are read from batcheventview. For recipes above phase-level recipes, unit batches require equipment acquisition and release events (unless equipment arbitration events are unavailable). For equipment arbitration events, the interface reads the [AcquireTime] and [ReleaseTime] values from batchequipmentview. For phase states, the start of a state ends any preceding state (in other words, there are no explicit end events for phase states.)

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

SQL UBR Start and Stop Events

Recipe Level	Database View	Start time column	End time column	Remarks
Batch	batchrecipeview	[ActivateTime]	[DeactivateTime]	For "procedure" events
Unit batch	batchrecipeview	[StartTime]	[EndTime]	Plus [AcquireTime] and [ReleaseTime] from batchequipmentview
Operation	batchrecipeview	[StartTime]	[EndTime]	
Phase	batchrecipeview	[StartTime]	[EndTime]	
Phase state	brecipestatechangeview or batcheventview	[StartTime]	[EndTime]	[EventType] field = "State Change" and [EventDescr] field containing the substring

To detect the start and end of levels when the UBR option is enabled, the interface queries the following views:

- batchrecipeview
- batchequipmentview
- brecipestatechangeview
- batcheventview

For operation- and phase-level recipes, the interface creates the parent levels (unit batch and batch).

SQL Query Details

The interface retrieves history from the DVHisDB database, primarily using a query that returns the union of the following tables:

- bactivestepchangeevent
- bmaterialchargerequestevent
- bmaterialchargeevent
- bpausestatusevent
- bequipmentselectionevent
- bphaselinkpermissiveevent
- brecipemodechangeevent
- brecipemodecommandevent
- brecipestatechangeevent
- brecipestatecommandevent
- brecipevaluechangeevent
- brecipevaluerequestevent
- breportevent
- btextmessageevent

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

- bphasebatchrequestevent
- brecipecomment
- unhandledbatchmsg

In addition, the interface retrieves batch-related data from the following views:

View or Table	Contents Queried For
batchview	uniqueid, batchid, start time, end time, product, uniqueid and archived flag with new archived database name for all batches.
brecipestatechangeview	State change events, which are used by default to detect start and end of recipe levels.
batchrecipeview	Recipe data, equipment linkage, start and end time for each object.
batchequipmentview	Equipment arbitration events.
batcheventview	Batch-associated data. Queried only if **Use original batch event view** (UOBEV) option is enabled. The [DESCRIPT] column is parsed for [PVAL] and [EU].
localevars	Used to convert local start and end times with DST offsets to GMT time and then to UTC seconds.

To detect start and end events, the interface queries the following views by default:

- brecipestatechangeview
- batcheventview
- batchequipmentview

Batch event start and end times are read from batcheventview. For recipes above phase-level recipes, unit batches require equipment acquisition and release events (unless equipment arbitration events are unavailable). For equipment arbitration events, the interface reads the [AcquireTime] and [ReleaseTime] values from batchequipmentview. For phase states, the start of a state ends any preceding state (in other words, there are no explicit end events for phase states.)

To detect the start and end of levels when the UBR option is enabled, the interface queries the following views:

- batchrecipeview
- batchequipmentview
- brecipestatechangeview
- batcheventview

Template placeholders

The following sections list placeholders provided by the interface for defining template settings.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

DeltaV Batch Historian

Placeholder	Relationship to Source Column	Source Column	View or Union of tables[1]
[AREA]	equivalent to	area	batcheventview
[BATCHID]	equivalent to	batchID	batcheventview
[COMMENT]	Paragraph	userComment	Union of tables
[DESCRIPT]	equivalent to	eventdescr	batcheventview
[EU]	Paragraph	engUnit	Union of tables
[EVENT] or [PARAMETER]	equivalent to	eventType	batcheventview
[OPERATION]	determined from equivalent to	action	batcheventview
[PHASE]	determined from equivalent to	action	batcheventview
[PHASEMODULE]	equivalent to	phasemodule	batcheventview
[PROCEDURE]	determined from equivalent to	action	batcheventview
[PROCESSCELL]	equivalent to	processcell	batcheventview
[PVAL] or [VALUE]			Union of tables[2]
[TIME]	n/a	n/a	n/a
[RECIPESTEP]	n/a	n/a	n/a
[RECIPETYPE]	n/a	n/a	n/a
[UNIQUEID]	equivalent to	uniqueid	batcheventview
[UNIT]	equivalent to	unit	batcheventview
[UNITPROCEDURE]	determined from equivalent to	action	batcheventview
[USERID] or [USER]	equivalent to	userName	batcheventview

[1]: Named view or union of tables as described in the topic: Emerson start and stop events

[2]: Value appropriate to each table in the union of tables, for example: the value column of the BReportEvent table.

For the **VALUE** setting and for **NAME** in property templates, the additional placeholders [TAG] and [TIME] are supported.

DeltaV Event Chronicle (Alarms & Events Historian) Placeholders

- [AREA]
- [ATTRIBUTE]
- [CATEGORY]
- [DESC1]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Interface for Emerson DeltaV Batch

- [DESC2]
- [EVENT]
- [LEVEL]
- [MODULE]
- [MODULEDESC]
- [NODE]
- [STATE]
- [TIME]
- [UNIT] (If [MODULEDESC] is "Unit Module", the [UNIT] placeholder contains the module name. Otherwise it contains the unit name.)

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

PI Event Frames Interface Manager

The PI Event Frames Interface Manager is a configuration tool that enables you to configure settings for batch interfaces. To configure settings using this tool, go to the **Interface Selection** tab and select the interface instance you want to configure, or add a new instance. On the remaining tabs, configure the settings for generating batch data. When you are done, go to the **Save Settings** tab to save your entries. To start the interface service with PI Event Frames Interface Manager, click **Start Interface Service** on the **Service Configuration** tab. To stop the interface service click **Stop Interface Service**.

The following sections describe the settings on the tabs of the PI Event Frames Interface Manager configuration tool.

Server Information tab

The **Server Information** tab is where you configure the PI Data server and PI Asset server that you intend to use with the interface instance. The interface can generate either batches in the PI Batch Database or event frames in the PI Asset Framework.

PI Data server (/HOST) settings

The PI Data server specified under PI Data server specifies the server to which the interface sends batch data. *Host* is the IP address or fully-qualified domain name of the PI Data server or PI Collective. If the server that you want to use is not listed in the drop-down menu, add it to the known servers table using the About PI-SDK utility.

If you are using a PI Data Archive version prior to 3.4.380.36, select **Use explicit log** in to specify a **PI user name** and **PI password**.

Note: When you use an explicit login to authenticate with PI Data server, PI Interface for Emerson DeltaV Batch stores the credentials you enter in Windows Credential Manager. For details, see Windows Credential Manager for authentication.

If you are using a PI Data Archive version 3.4.380.36 and higher, use Windows Integrated Security for authentication. Ensure that the Windows account that runs the interface has sufficient permissions on the PI Data server to write data to PI points.

Note: You must then configure a trust that permits access for the user that runs the interface service.

PI Asset server settings

In this section you configure PI Asset Framework configuration settings.

Select **Create event frames in PI Asset Framework** to create event frames in PI Asset Framework, instead of creating batches in the PI Batch Database.

The PI Asset server and PI Asset Framework database are displayed in the **Host and Database** (/AFHOST and /AFDATABASE) field. Click **Select Asset server** to open the Select Database window to choose an alternative server.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Select the **Enable Batch Interface Element** check box to create a new element that contains the Interface Health Tag information.

Interface Health Tags collect batch statistics for the interface, such items as process time for unirecords or the number of unirecords in a batch.

Note: The naming convention used for a Batch Interface Element is *(InterfaceInstanceName)_(InterfaceID)*_BatchStatistics.

Click **Select Parent Element** to open a dialog box to search for a parent to host the new element. After locating the parent, select OK to return to the Server Information tab.

Note: A Batch Interface Element can be created in a separate Asset Framework database.

Under **Enable Auto Check In**, select from the following options:

- **None**: The interface will do nothing with checked out Event Frames on a restart/reconnection.
- **Check In**: The interface will check in any checked out Event Frames on a restart/reconnection.
- **Rollback**: The interface will undo any checked out Event Frames on a restart/reconnection.

Note: If the same user account is running multiple batch interfaces, our recommendation is to set **Enable Auto Check In** to None. The OSIsoft best practice is to have a separate service account for each batch interface.

If you are not using Windows Integrated Security for authentication, check **User explicit login for PI Asset Framework** and enter the **Windows account** and **Password** for the Windows user account that you intend to use to connect to PI Asset Framework.

Note: When you use an explicit login to authenticate with PI Asset Framework, PI Interface for Emerson DeltaV Batch stores the credentials you enter in Windows Credential Manager. For details, see Windows Credential Manager for authentication.

Windows Credential Manager for authentication

As a security best practice, PI Interface for Emerson DeltaV Batch stores any credentials that you enter during interface configuration to Windows Credential Manager. PI Interface for Emerson DeltaV Batch uses these credentials to authenticate with:

- PI Data server
- PI Asset Framework
- The batch interface data source

Storage of credentials within Windows Credential Manager benefits you by:

- Caching credentials for future authentication, so that you do not have to re-enter them.
- Removing any reference to passwords within the interface instance .ini file.
- Encrypting the credentials.

Tip: For more information on where to configure credentials that PI Interface for Emerson DeltaV Batch uses for authentication, see:

- Server Information tab
- Source tab

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Credential creation

When you configure an interface using the PI Event Frames Interface Manager (see links in tip above), PI Interface for Emerson DeltaV Batch temporarily writes the credentials that you enter to the applicable .ini file. These settings remain in the .ini file until you start the interface.

After you start PI Interface for Emerson DeltaV Batch, it makes the following updates:

- All credentials are saved to Windows Credential Manager.
- Within the interface .ini file, any reference to passwords are removed.

Credential storage

After you start PI Interface for Emerson DeltaV Batch, the credentials that it uses for authentication with PI components and data sources are listed in Windows Credential Manager.

The following table lists credentials for PI Interface for Emerson DeltaV Batch:

Credentials	Name
PI Data server	EMDVB_X_PI
PI Asset Framework	EMDVB_X_AF
Batch interface data source	EMDVB_X_Source_Y

Notes:

- X is a placeholder for an interface instance **Service ID**.
- Y is a placeholder for a data source instance number.

To update credentials

If you need to update the credentials that the interface uses for authentication, edit the configuration settings for the instance using PI Event Frames Interface Manager.

1. From **Interface Selection**, select the applicable PI Interface for Emerson DeltaV Batch instance.
2. From **Server Information**, update the credentials used to authenticate with PI Data server or PI Asset Framework.
3. From **Source**, update the credentials used to authenticate with a data source.
4. Click **Save Settings**.

Result: The credentials are updated within the instance configuration. Restart the interface to update the credentials within Windows Credentials Manager.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Source tab

Configure the data sources that you want the interface to read. To add a data source, right-click the top **Sources** node and choose the **Add** option for the type of source you are configuring. Complete the form to enter authentication information for the data source.

Note: PI Interface for Emerson DeltaV Batch stores the data source credentials you enter in Windows Credential Manager. For details, see Windows Credential Manager for authentication.

You can configure the interface to collect real-time data from an OPC Alarms and Events server. If the interface loses connectivity to the specified OPC Alarms and Events server, it switches to the specified SQL server and continues to collect data until it is able to reconnect to the OPC Alarms and Events server.

To configure a DeltaV Event Chronicle (alarms & events) data source, you must specify mappings that ensure that process cell data is recorded correctly, because DeltaV Event Chronicle does not emit process cell information.

To use explicit logins for an SQL data source, ensure that the user running the interface and the user configuring the interface are the same user.

Templates tab

Templates map data from the data source to PI tags, batches and event frames. When defining templates, you specify the data to be written and the events that trigger the update. To define templates using PI Event Frame Interface Manager, go to the **Templates** page and navigate to the desired type of template. You can configure templates that map the source data to string, integer or float data types. For each type of template, you specify triggers, which are conditions that active the template.

You can define the following types of templates:

- **Tag:** Creates and updates PI tags, specifying how they are named and what data is written to them.
- **Property:** Maps data to batch properties in the PI Batch Database or event frame attributes in PI AF.
- **Recipe:** Defines the information stored and the naming convention used at each level in the generated batch hierarchy.
- **Alarm tag:** Writes data to a PI tag when the data source raises an alarm.

The following sections provide details about the specific types of templates.

Alarm tag templates

Alarm tag templates allow you to configure the event notifications you will receive.

Configuration

Index

Assigns a unique numeric identifier for the template.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Name

Specifies how the alarm tag is named.

Click the **Name** field to activate the **Add Placeholder** button. Click the button to open the **Build a Placeholder** window. There you can select and add a placeholder and select from the following Advanced parsing features:

- **Add value**: Select to add and specify placeholder values. Click **Add** to save your changes.
- **Add substring parsing**: Select and specify substring values. Click **Add** to save your changes.
  - Select **Left bound** and either **LBE** or **LBI** from the dropdown menu to define the left boundary of the substring.
  - Select **Right bound** and either **RBE** or **RBI** to define the right boundary of the substring.
  - Select **Delimiter** then provide a name and a count to narrow the resulting substring.

Value

Specifies the value to be recorded. As with the **Name** field, click the **Value** field to activate the **Add Placeholder** button. You can edit placeholders to include values from the data emitted by the data source.

Data type

Specifies the PI data type of the value from the following options:

- String
- Integer
- Float
- Auto
- Float or string

Translate

Click the checkbox to activate the **Translate** button, which you can click to open the **Translate** window. There you can add and remove English Text and Foreign Text.

Advanced features

Descriptor

Populates the tag's Descriptor field.

Engineering units

Unit of measurement.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Alias

Use the drop down menu to select from the following:

- **Default**: Defaults to alias name and filepath.
- **Custom alias name**: Provide a custom alias using one of the following format examples:
  - Tag(#)alias = ABC\def |test[module]
  - Tag(#)alias = $[Area] |[module] ALARM alias
- **Disable alias creation**: No alias is created.

Annotation

Use the field provided to annotate the alarm tag.

You have the option of selecting the **Use a NameValues collection for annotation** or **Remove annotation from tag value**.

Triggers

To specify the events that initiate updates to tags or properties, you define triggers. You can define triggers for events read from the data source and for batch-related events raised by the PI System itself (PI Events). To define triggers using PI Event Frame Interface Manager, go to the Templates page, navigate to the desired tag or property template, and click the Triggers tab.

If you omit triggers, the target is updated by every event. If you specify multiple conditions in a single trigger, data is written only when all conditions are met (logical AND). If you define multiple triggers, data is written when any one of the conditions is met (logical OR).

In the following example, the template is triggered when the PI System records the start of a batch:

```
[Event,value="PIEVENT"] [Descript, value="BATCH"] [Pval, value="START"]
```

Property templates

Property templates enable you to configure how data from the data source is written to event frame attributes or batch database properties. Each interface provides a set of default properties, which contain commonly-desired data. To add these properties, right-click the **Property Templates** node and choose **Add Default Templates**.

Configuration

Index

Assigns a unique numeric identifier for the template.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Name

Specifies how the property is named. You can use placeholders to include values emitted by the data source. By default, properties reside under a parent recipe property. To specify a location in the hierarchy where you want the property to reside, define the name as a path, delimited using backslashes. For example, to specify that all properties generated using this template are located under a parent property named "Materials" plus the contents of the **Event** field, specify the name as follows: `Materials\[Event]`. To create the property at the root level, precede the name with a dollar sign: `$\[Event]`.

Value

Specifies the value to be recorded. Use placeholders to include values from the data emitted by the data source.

Create property

Specifies the level at which the property is to be created. Options are as follows:

- **At recipe level**: By default, properties are created at the recipe level corresponding to the level in the data source from which the data was read.
- **Under root node**: Create properties at the top level of the batch or event frame hierarchy, regardless of the level at which they originated.
- **Under UniqueID node**: (Batch database only, not event frames) Create properties under the batch's UniqueID node.

Data type

PI data type of the value. Note that if the incoming value is incompatible with the specified type, the event is not processed, and an error is logged.

Translate

Maps text from the data source to the text that you want to record in the PI System.

UOM

Unit of measure to be used to store value, if different from unit provided by data source.

Advanced features

Descriptor

Populates the tag's Descriptor field.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Engineering units

Unit of measurement.

Category

Specifies the PI AF category to be associated with the value.

Recipe templates

Recipe templates enable you to configure the names that are assigned and the data that is stored at each level of the batch hierarchy. A set of default recipe templates are is included with each batch interface. To display and override the default recipe templates, right-click the **Recipe Templates** node and choose **Add Default Templates**. The settings for recipe templates are as follows.

Configuration

Index

Specifies the level in recipe hierarchy as follows:

Level	Index	PI object	Default
Procedure	1	PIBatch Recipe	Recipe[1].Name=[Procedure]
Unit Procedure	2	PIUnitBatch Procedure	Recipe[2].Name = [UnitProcedure]
Operation	3	PISubBatch Name field	Recipe[3].Name=[Operation]
Phase	4	PISubBatch Name field	Recipe[4].Name=[Phase]
Phase State	5	PISubBatch Name field	Recipe[5].Name=[PhaseState]
Phase Step	6	PISubBatch Name field	Recipe[6].Name=[PhaseStep]

Name

Defines the naming convention used by the interface to assign names to batch events. For event frames, this template modifies the Recipe AF attribute. For the Batch database, this template modifies the PIBatch name.

For example: abc_[Procedure] If the incoming event's [Procedure] field contains "Test", the resulting procedure **Recipe** field is "abc_Test".

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Value

Note: The **Value**, **Create property**, **Data type**, and **Translate** options are only available for recipe template properties.

Specifies the value to be recorded. Use placeholders to include values from the data emitted by the data source.

Create property

Specifies the level at which the property is to be created. Options are as follows:

- **At recipe level**: By default, properties are created at the recipe level corresponding to the level in the data source from which the data was read.
- **Under root node**: Create properties at the top level of the batch or event frame hierarchy, regardless of the level at which they originated.
- **Under UniqueID node**: (Batch database only, not event frames) Create properties under the batch's UniqueID node.

Data type

PI data type of the value. Note that if the incoming value is incompatible with the specified type, the event is not processed, and an error is logged.

Translate

Maps text from the data source to the text that you want to record in the PI System.

Specifies the value to be recorded. Use placeholders to include values from the data emitted by the data source.

Advanced features

Batch ID

Configures the batch ID of the particular recipe object, overriding the incoming (default) batch ID. If you override the batch ID for the procedure, the batch ID is propagated to the child unit batches Batch ID field. For event frames, this template modifies the event frame name.

In the Procedure level recipe template, the interface uses the Batch ID field for the name of the Event Frame if it is the top level of the batch hierarchy. If the Batch ID field is empty, an error is returned during recipe template processing. If you are using a truncated Batch ID and the Batch ID field is not empty, the interface uses the Batch ID that matches the mask provided by the BIDM for the Batch ID. BIDM only shows noticeable difference in Event Frames that use the Batch ID field for their names.

In the Unit Procedure level recipe template, the interface uses the Name field for the name of the Event Frame based on the Batch Recipe (for example, level is s88_UnitBatch). The Batch ID field is an attribute/property in the Event Frame.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

In the Operation level recipe template and all other sub batches, the interface uses the Name field for the name of the Event Frame. The Event Frame does not store the Batch ID field.

Note: If you use a recipe template to set the batch ID, the recipe template overrides any batch ID mask you might have configured to enable merging of batches.

Module/Element Path

Specifies the location in the PI module or AF element hierarchy where the unit or phase module resides. Valid for unit procedure (level 2) and phase (level 4).

Product

Specifies the product of the particular recipe object. Supports the procedure and unit procedure [Product] fields, which must be present in the source event that creates the batch. If a product trigger is not defined, this template is populated based on the data in the event that creates the particular Recipe object.

Product Trigger

Populates the [Product] field of the particular recipe object after the object is created, which is useful if the product is defined in a separate event.

For example: `[Parameter, Value="Recipe Header"] [Descript, value="Product Name"]`.

Event Frame Template

(Event frames only) Specifies the AF template to be used to create event frames for this recipe.

Category

(Event frames only) Specifies the AF category to be applied to the event frame.

Merge same named objects under parent

Combines identically named child objects.

Recipe templates default properties

Batch Level

The following are the recipe template default values for the BATCH level:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Template	Default
Name	[Procedure]
Value	[BatchID]
Product	[pval]
Product Trigger	[Event,value="Recipe Header"] [descript,value="Product Code"])
Category	OSIBatch
Template	Procedure

Default Properties

[RECIPE]

The index in the recipe template heirarchy corresponds to the RECIPE item default property (1).

```
  Recipe[1].DEFAULTPROPERTY[1].NAME = Recipe
  Recipe[1].DEFAULTPROPERTY[1].VALUE = [PROCEDURE|UNITPROCEDURE|OPERATION|
Phase|PhaseState|PhaseStep]
```

[PRODUCT]

The index in the recipe template heirarchy corresponds to the PRODUCT item default property (2).

```
  Product[2].DEFAULTPROPERTY[2].NAME = Product
  Product[2].DEFAULTPROPERTY[2].VALUE = [pval]
  Product[2].DEFAULTPROPERTY[2].TRIGGER = [Event,value="Recipe Header"][de
script,value="Product Code"]
```

Unit Batch Level

The following are the recipe template default values for the UNIT BATCH level:

Template	Default
Name	[UnitProcedure]
Value	[BatchID]
Product	[pval]
Product Trigger	[Event,value="Recipe Header"] [descript,value="Product Code"])
SearchByStartTime	true
SearchByEndTime	true
Category	OSIBatch
Template	UnitProcedure
Module Path	[AREA]\[PROCESSCELL]\[UNIT])

Note - Unitbatches can have the same name when SearchByStartTime and SearchByEndTime equal true.

Default Properties

[PROCEDURE] = 3

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
   .NAME = PROCEDURE
   .VALUE = [UNITPROCEDURE|OPERATION|Phase|PhaseState|PhaseStep]
```

[PRODUCT] = 2

```
   .NAME = Product
   .VALUE = [pval]
   .TRIGGER = [Event,value=\"Recipe Header\"][descript,value=\"Product Code
\"]
```

[RECIPE] = 1

```
   .NAME = BatchID
   .VALUE = BatchID
```

Operation Level

The following are the recipe template default values for the OPERATION level:

Template	Default
Name	[Operation]
Category	OSIBatch
Template	UnitProcedure

Phase Level

The following are the recipe template default values for the PHASE level:

Template	Default
Name	[Phase]
Category	OSIBatch
Template	Phase
Module Path	[AREA]\[PROCESSCELL]\[UNIT]\[PHASEMODULE])

Phase State Level

The following are the recipe template default values for the PHASE STATE level:

Template	Default
Name	[PhaseState]
SearchDirection	Reverse
SearchByStartTime	true
SearchByEndTime	true
Category	OSIBatch
Template	PhaseState

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Phase Step Level

The following are the recipe template default values for the PHASE STEP level:

Template	Default
Name	[PhaseStep]
Merge	false
SearchDirection	Reverse // search in Reverse (default: always FORWARD)
Category	OSIBatch
Template	PhaseState

Tag templates

Tag templates create and update PI tags when events are read from the data source. Alarm tag templates create and update PI tags based on alarms raised by Emerson DeltaV Event Chronicle (alarms and events historian). If you define one or more triggers, the target PI tag is updated only when the specified events occur. If you do not define any triggers, every event from the data source triggers an update of the target PI tag.

Tag aliases enable you to associate PI tags with equipment assets (PI units or AF elements). All batch interfaces support the creation of aliases at the unit batch and phase level. In addition, some batch interfaces permit you to define aliases at any level of the batch hierarchy by providing level-specific placeholders. For details, refer to the interface-specific section in this guide.

Tag configuration

Attribute	Definition
Index	Specify a unique numeric identifier for the entry.
Name	Configures how the tag is named. To include data from the data source, use placeholders.
Value	The value to be written to the PI tag. To include data from the data source, use placeholders.
Type	The data type to be used to write the value to the tag. Note that if the incoming value is incompatible with the tag type, the event is not processed, and an error is logged.
Translate	Maps text from the data source to the text that you want to record in the PI System.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Advanced features (attributes)

Attribute	Purpose
Descriptor	Populates the descriptor field for the target PI tag.
Engineering units	The engineering units for the data.
Unit Alias	Specifies an alias to be recorded under the Aliases node in the corresponding PIUnit or AF element.
Phase Alias	Specifies an alias to be recorded under the Phases > Aliases node in the corresponding PIUnit or element.
Annotation	Specifies the annotation to be associated with the event. The result is stored in a PI tag as a string.
Use the NamedValues collection for annotations	Store annotations in a PI tag as a name-value collection. The name is derived from the event sent by the data source.
Remove annotations from tag values	Store values without annotations.

Filters tab

You use the **Filters** tab to configure the recipes, units, phases, or phase states to be excluded from processing by the interface.

- **Skip Phases (/SKIPPHASES)**

  The interface ignores any event that contains the specified phases in the [Phase] or [Phasemodule] column.
- **Skip Units (/SKIPUNITS)**

  The interface ignores any event containing the specified units in the [Unit] column.
- **Skip Recipes (/SKIPRECIPES)**

  The interface ignores any event containing the specified recipe in the appropriate column ([Procedure] for a procedure recipe, [UnitProcedure] for a unitprocedure recipe, and so on.)
- **Exclude Phase States (/EXCLUDESTATES**

  The interface ignores phase state events that contain the specified phase states in the [PhaseState] column.

Time Settings tab

You use the **Time Settings** tab to configure the settings that control how the interface handles loss of connectivity and how it processes data.

Query time settings

Scan (/SCAN=)

Specifies how frequently the interface scans the data source.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Cache time (/CACHETIME=)

Specifies how long completed events are retained in memory. Default is one day. Specify the maximum duration expected between events that need to be merged, plus any desired margin of safety. The value can be specified as whole day or fraction of a day. For example, to release completed batches when their end time is less than 7 days and 12 hours from current time, specify the following cache time setting:

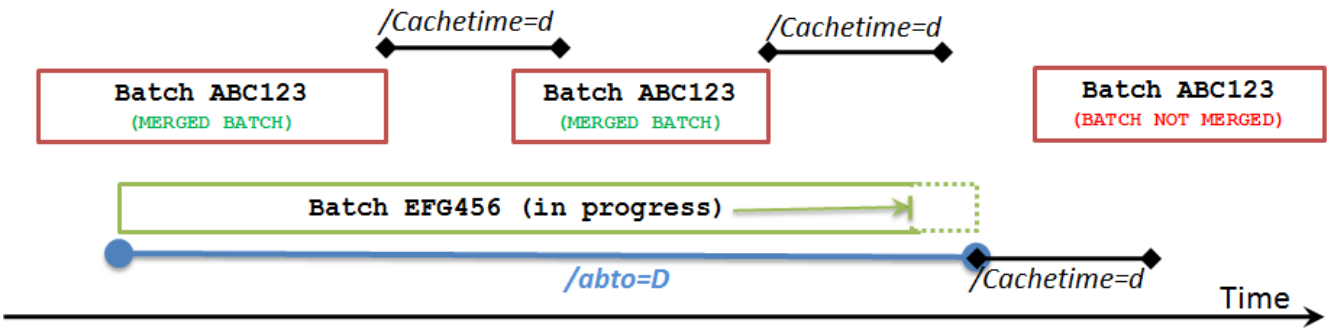
```
CACHETIME=7.5 days
```

Abandoned batch timeout (/ABTO=)

The timeout has 2 functions:

1. In history recovery, if a batch starts less than the recovery start time minus the abandoned batch timeout it is not recovered. For example, if a batch is open in recovery start time, and the start time is June 1, 2019 and we have a 30-day abandoned batch time, and the batch you found is on May 1, 2019, it will not be recovered.
2. In real time mode, if a batch has not received a start or end time in the hierarchy - meaning in the top level of the event frame, or any of the children - and if the last updated time you get is less than the cache time minus the abandoned batch timeout, it will be considered abandoned. If July 1 is real time and cache time is one day, and our abandoned batch timeout is 30 days, and our last start time was April 29, it will be thrown from cache as abandoned.

The following graphic illustrates the timeout logic:



Maximum query time frame (/MAXQTF=)

To help manage workload and memory usage, defines the maximum time frame for queries. For example, if you specify 30 days and the interface queries for one year's worth of data, the interface issues 12 one-month queries rather than one (large) one-year query.

Maximum stop time (/MAXSTOPTIME=)

Specifies the maximum time allowed for the interface to properly shutdown. If the shutdown process takes longer than the specified time, the interface is forced to terminate immediately. The default value is 120 seconds.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Query Buffer Delay (/DELAYBUFFER=)

Specifies the amount of time (in minutes) subtracted from the current time when calculating the Next Query End Time. This configuration allows the system to catch recently closed batches that were missing from the latest database call.

The default value is two minutes, but you can configure the amount of time to a maximum of 10 minutes. To disable the configuration, set the /DELAYBUFFER to zero (0) in the INI file.

This configuration is used in both real-time and history recovery mode but only applies to interface configurations where the end time for data recovery (/ret) is not defined.

Use local time stamps to process incoming events (/TS)

Specifies whether timestamps from an SQL data source are interpreted as local time or GMT. By default, timestamps are interpreted as **local time**.

PI connection settings

PI connection timeout (/PICONNTO=)

Override the default PI Data server Connection TimeOut property.

PI data access timeout (/PIDATO=)

Override the default PI Data server Data Access TimeOut property.

Retry (/RETRY=)

Specifies how often the interface retries failed attempts to write data. The default is 60 seconds.

Retry timeout (/RETRYTO=)

Specifies how long the interface continues retrying an attempt to write data. To avoid data loss, set to 0 (default, no timeout).

Batch Setup tab

The settings on this tab configure batch generation as described in the following table.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Batch settings (Command line parameter settings)

Report as step (/RAS)

Use the "Report" event to create phase steps under active phase states.

Alternate PI module path (/SMP)

Specifies an alternate PI Module path or PI AF element path for a particular equipment hierarchy.

Disable arbitration (/NOARBITRATION)

Create unit batches based solely on source batch recipe data. For use when the source Batch Executive System (BES) provides batch data without equipment arbitration data.

Disable arbitration counters (/DAC)

Directs the interface to release a unit on the first resource release event even if the number of acquire events is higher than number of release events.

Use batch recipe (/UBR)

Default setting: /UBR = false

/UBR can be set in the .ini file, from the command line, or by using the PI Event Frames Interface Manager / batch Setup tab.

If /UBR = true the interface will use SYSTEM MESSAGE to control the start and end of event frames. Example System Messages are BEGIN OF BATCH, END OF BATCH, UNIT PROCEDURE STARTED, and UNIT PROCEDURE ENDED

If /UBR = false the interface will use STATE CHANGE to control the start and end of event frames. Example State Change messages are RUNNING, REMOVED, ABORTED, COMPLETE, STOPPED, and ABANDON. The interface will combine the state change with the recipe (Batch, UnitProcedure, Operation, Phase) to determine which recipe step has changed state.

Use original batch event view (/UOBEV)

Directs the interface to use the original batch event view. By default the interface queries multiple tables to retrieve data for batch-associated events.

True batch start end (/TBSE)

Use top level recipe start/end events for creating batch objects.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Allow deferred units (/ADU)

Enable the creation of unit batches for recipes in units that are allocated at the phase level rather than the unit batch level.

Disable auto module creation for A&E data source (/DAMCAE)

Ignore events from a DeltaV Event Chronicle (alarms & events) data source when creating or checking PI Module Database objects.

Remove trailing index from recipe fields (/RTI)

Remove trailing index from Recipe fields. Applicable to Procedure, Unit Procedure and Operation Recipe fields. Emerson EVT data source only.

Merge settings (Command line parameter settings)

Merge operation (/MOP)

Merge identically-named operations under the same parent unit procedure.

Merge unit procedures (/MUP)

Merge identically-named sequential unit procedures running on the same unit into a single unit procedure.

Merge multiple source batches with same batch ID into one batch (/MERGE)

Enable merging of multiple source batches with same ID. The original data for each batch merged is stored in PI properties under a node named using the ID of the original batch.

Batch ID mask (/BIDM)

Override the incoming batch ID.

Truncate batch ID (/TBID)

Use the truncated batch ID in the batch ID field of unit procedures.

Link to child eventframes(/WRITELINK)

Combine event frames from different interface instances. For an MES controlling one or more BES systems, configure Link to child event frames on the MES interface and configure an interface instance for each BES, specifying the same linkage element in the BES Link from parent event frames setting. The BES interface

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

instances will then create event frame references under the MES event frames that refer to the BES event frames. For Emerson Syncade systems, the AutomationBatchID field must match the batchID of the batch created by the BES.

For a BES interface controlling one or more MES systems, configure Link from parent event frames on the MES interface and configure an interface instance for each BES, specifying the same linkage element in the BES Link to child event frames setting. The MES interface will then create event frame references under the BES event frames that refer to the MES event frames. Link templates must also be configured to define which events specify a link.

Link from parent event frames (/READLINK)

Combine event frames from different interface instances. For an MES controlling one or more BESsystems,configure Link from parent event frames on each BES instance, specifying the same linkage element in the MES Link to child event frames setting. The BES interface instances will then create event frame references under the MES event frames that refer to the BES event frames. For Emerson Syncade systems, the AutomationBatchID field must match the batchID of the batch created by the BES.

For a BES controlling Lione or more MES systems, configure Link from parent event frames on each MES instance, specifying the same linkage element in the BES Link to child event frames setting. The MES interface instances will then create event frame references under the BES event frames that referto the MES event frames. Link templates must also be configured to define which events specify a link.

Operational Settings tab

Modify the settings on the Operational Settings tab to configure the interface mode and other related settings.

Operational Settings

Realtime mode (/MODE=)

Interface modes are selected from the drop-down menu. Options are described in the table below.

For the Recovery, Statistics, Preprocess, and Delete options, you can identify a **Start Time** and an **End Time**. Click the calendar icon associated with each field to select dates and times.

Mode	Description
Realtime (default)	Scan data source to collect data in realtime.
Recovery	Scan data source and generate or correct events accordingly. Always starts in recovery mode, then switches to realtime mode.
Statistics	Compare data source history against events and report results without updating any data.
Preprocess	Source data is written to the PI Data archives with timestamps earlier than those written to the primary PI archive.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Mode	Description
	Note: This mode requires that you use the offline archive utility to reprocess older archives. This is done after completion when the interface stops. **Note:** This mode requires that you use the offline archive utility to reprocess older archives. This is done after completion when the interface stops.
Delete	Delete events for a specified period. **Note:** Selecting "Delete mode" limits you to specifying a Start Time only. In the GUI, the End Time is always specified with "*" asterisk symbol, a dereference operator pointed to by the variable's value. The interface allows for "spot deletion", meaning the deletion of event frames from within a specific time window. For example, you may have a scenario where you have received "junk data" from previous days. Spot deletion allows you to delete event frames back to the date when you began receiving unusable data. The GUI does not provide the means to specify both ends of the time window that you wish to delete. That can be done through edits to the configuration file. An RST for Delete mode can still be specified in the initialization file, but the GUI will always disable spot deletion.

Perform one scan then stop (/SINGLERUN)

The interface performs one scan of active points, then exits.

Print result of first scan to file (/PRINT=)

Print the results of the first scan to the specified text file. The results include the event frame hierarchy tree, the tag list, and the equipment tree. This parameter is designed primarily for troubleshooting and configuration testing when the interface is run in statistics mode. Enter a name for the scan in the field provided (for example, `C:\Users\jdoe\Documents\PIWPASXBatch1_stat.dat`). You have the option of clicking **Select file** to browse to the file path.

Local debug messages (/DB=<#>)

Specifies level of detail for logging as follows:

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- 0: Log errors and warnings (default)
- 1: Log errors, warnings and major successes
- 2: Log all messages
- 3: Log all messages including Unirecord diagnostic messages

Numeric settings (/NS=)

Configures how numeric values are formatted by the interface, to enable the interface to properly interpret numeric values based on the machine's regional setting or a user-specified language. Default is "English (United States)".

Interface ID (/ID=x)

Specifies the numeric interface instance identifier (maximum nine digits). To detect PI points maintained by the interface instance, the interface matches this setting against the value in the points' Location1 attribute.

Point source (/PS=x)

Point source for the interface instance. Point source is not case sensitive. Corresponds to the PointSource attribute of individual PI points. The interface loads PI points with the same point source.

Failover settings

Failover tag (/FAILOVERTAG=x)

The failover tag coordinates failover by using a PI tag. You must create the PI tag on the PI Server before creating a failover tag. To create a PI tag, refer to the System Management Tools Point Builder. When creating a PI tag, you should use *string* for the point type.

To create a failover tag, select to open the Tag search window. In the Tag Mask field, enter the name of the tag.

The created tag displays in the Failover tag field of the PI Event Frames Interface Manager.

Failover identifier (/FAILOVERID=x)

A unique identifier for this interface instance, used to coordinate which instance is primary.

Failover swap time (/SWAPTIME=)

Select to specify how long an interface can be inactive before failover to another instance occurs. The default is 300 seconds.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Security settings

Specify point security (/PTSEC=x)

Override the default point security created by the interface. Click **Change** to open the Change security window. There you can select from or add to the options listed, and specify read/write authorizations for each. Click **OK** to change point security.

Specify data security (/DATASEC=x)

Override the default data security of PI points created by the interface. Click Change to open the Change security window.

Monitor Tag Setup tab

From the **Monitor Tag Setup** tab, you can disable monitor tags for the PI Interface for Emerson DeltaV Batch that you do not use. Each batch interface supports approximately 40 tags, with each tag licensed in groups. Disabling unused tags reduces license consumption. Select each tag that you want to disable.

Monitor tag options

To enable or disable tags in bulk, use the **Select all tags** and **Unselect all tags** options at the top of the window.

Option	Description
Select all tags	Disables all monitor tags listed.
Unselect all tags	Activates all monitor tags previously disabled.
Monitor tag time window in days	Most monitor tags show a sum of events over a number of days. This setting allows the size of the window to be adjusted. The default value is one day.

Monitor tag reference

The following table lists each monitor tag that you can disable from the **Monitor Tag Setup** tab.

Monitor Tag	Point Type	Archiving[1]	Description
BatchStatus	Int32	true	Status of interface data collection.
BatchProcessorStatus	Int32	true	Status of interface data processing.
BatchListCount	Int32	false	Number of objects in the cache.
EquipmentCount	Int32	false	Number of equipment objects in the cache.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Monitor Tag	Point Type	Archiving[1]	Description
TagListCount	Int32	false	Number of cached tags.
TagAEListCount	Int32	false	Number of alarm and event tags.
EventReadCount	Int32	false	Number of events processed.
ErrorCount	Int32	false	Global error counter.
SourceUnitCount	Int32	false	Number of unit modules found on source.
PIUnitCount	Int32	false	Number of unit modules found on and added to destination.
SourcePhaseModCount	Int32	false	Number of phase modules found on source.
PIPhaseModCount	Int32	false	Number of phase modules found on and added to destination.
SourceBatchCount	Int32	false	Number of batches found on source.
PIBatchCount	Int32	false	Number of batches found on and added to destination.
SourceUnitBatchCount	Int32	false	Number of unit batches found on source.
PIUnitBatchCount	Int32	false	Number of unit batches found on and added to destination.
SourceSubBatchCount	Int32	false	Number of sub batches found on source.
PISubBatchCount	Int32	false	Number of sub batches found on and added to destination.
SourcePropertyNodeCount	Int32	false	Number of property nodes found on source.
PIPropertyNodeCount	Int32	false	Number of property nodes found on and added to destination.
SourcePropertyEventCount	Int32	false	Number of property events found on source.
PIPropertyEventCount	Int32	false	Number of property events found on and added to destination.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Monitor Tag	Point Type	Archiving[1]	Description
SourceTagCount	Int32	false	Number of tags found on source.
PITagCount	Int32	false	Number of tags found on and added to destination.
SourceTagEventCount	Int32	false	Number of tag events found on source.
PITagEventCount	Int32	false	Number of tag events found on and added to destination.
SourceTagAliasCount	Int32	false	Number of tag aliases found on source.
PITagAliasCount	Int32	false	Number of tag aliases found on and added to destination.
CachedBatchCount	Int32	false	Number of batches in cache.
OpenBatchCount	Int32	false	Number of open batches in cache.
SourceReadTime	Float32	false	Time spent reading source data per iteration.
TagCacheTime	Float32	false	Time spent updating the tag cache per event.
BatchCacheTime	Float32	false	Time spent updating the batch cache per event.
EquipmentCacheTime	Float32	false	Time spent updating the equipment cache per event.
BatchSyncTime	Float32	false	Time spent synching with PI and AF per event list.
TagSyncTime	Float32	false	Time spent synching tags per event list.
EquipmentSyncTime	Float32	false	Time spent synching equipment per event list.
TotalTime	Float32	false	Total time spent caching and synching batches, tags, and equipment per event list.
WaitingForEquipmentUB	Int32	false	Number of unit batches waiting for unit defining events.

[1]: Indicates whether the point that is created archives past values.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Note: The following monitor tags cannot be disabled:

- DeviceStatus
- HeartBeat

Test configuration tab

You test the configuration settings in the Test Configuration tab.

Specify test settings as follows, then click **Run Test**, and check the output file for results.

Field	Description
Output File	The path to the text file where the test results are written.
Start Time	Start time for scanning
End Time	End time for scanning

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Command line parameter reference

To configure an interface, you use the PI Event Frames Interface Manager, which maintains the files that contain batch interface settings. This appendix describes the command line settings and is provided for troubleshooting purposes.

Note: To ensure that settings files are formatted correctly, always use PI Event Frames Interface Manager to configure settings. Do not edit settings files manually.

The following table is a compilation of the command line settings for all the OSIsoft batch framework interfaces. Some settings are specific to an interface. To list the settings supported by your interface, invoke its executable at the command line, specifying the -? flag.

Available command line parameters

The following headings describe each command line parameter available.

/abto =<#days>

(Optional) Specifies how long, in addition to the CACHETIME setting, the interface waits before closing a batch for which no end event has arrived. When an abandoned batch is closed, the interface uses the timestamp of its last event as the end time and logs an "Abandoned batch found" message. Default is 100 days, minimum is .042 days (approximately one hour), and maximum is 365 days.

/adu =[true | false]

(Optional) Enable the creation of unit batches for recipes in units that are allocated at the phase level rather than the unit batch level. By default, the interface requires the unit name to be present in the unit batch start event. When you enable /adu, the interface creates the unit batch and defers setting the unit name until the phase-level allocation event arrives.

/bidm =<list>

(Optional) Override the incoming batch ID by selecting a substring. Specify one or more masks composed of text and wildcards, to be used to compose the desired batch ID from the contents of the source BatchID field. The resulting batch ID is used for the Batch ID field of the top-level procedure and for the [BATCHID] placeholder. By default, the batch ID field in unit procedures contains the full batch ID from the data source. To use the truncated batch ID instead, configure the TBID setting.

Valid wildcards are as follows:

Wildcard	Description
#	Single digit numerical value, 0-9
@	Single alpha character, a-z, A-Z

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Wildcard	Description
?	Any single symbol
!	Repeat the previous mask symbol
*	Any set of symbols

Wildcard use example

If the incoming Batch ID is: `lot30112 / 90dev123 / 12345stp / ld567`

The following list shows example masks and resulting data.

Mask	Result
"######"	30112
"##@!"	90dev
"*###@!"	lot30112 / 90dev
"@@@@, #8dev4, #!"	30112

In the last example, the first and second masks do not match, so the third mask is used.

/cachetime=<days>

(Optional) Batches are cached in memory to enable the interface to capture events sent by the BES after the batch has closed. This setting specifies how long (in days) completed batches are cached. To specify fractions of a day, use decimal values. For example, /cachetime=7.5 releases completed batches when their end time is more than 7 days and 12 hours from the current time. The default value is 1 day, minimum is .042 days (approximately one hour), and maximum is 60 days.

/dac

(Optional) Disable arbitration counters: directs the interface to release a unit on the first resource release event even if the number of acquire events is higher than number of release events. By default, the interface requires the number of acquire and release events for a unit to be the same.

/damcae

(Optional) Ignore events from a DeltaV Event Chronicle (alarms & events) data source when creating or checking PI Module Database objects. If the module path defined for an AlarmTag[#].Alias entry contains the root node symbol ($), the interface checks the module path regardless of whether this option is enabled.

/datasec=<string>

(Optional) Specifies the security settings to be assigned to interface-generated tags. For PI Data Archive 3.4.375.99 or earlier, use owner, group, world format. Example: /datasec="o:rw g:r w:r" For PI Data Archive 3.4.380.36 or later, specify an access control list (ACL).

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Example:

```
/datasec="piadmin: A(r,w) | PIEngineers: A(r)"
```

/db =[#]

(Optional) Enabled debugging output:

- 0: Log only errors and warnings (default)
- 1: Log errors, warnings and major success messages
- 2: Log all messages (most verbose)

/dpretc

(Optional – event frames only) Disable propagation of referenced elements to children. By default, the interface propagates each event frame element reference to its children event frames.

/enabledmonitortags =<TAG>]

Indicates which monitor tags are enabled. Each enabled tag consumes a license, but disabled tags do not. You can enable multiple monitor tags by separating each one with a comma-separated list. For example:

```
/enabledmonitortags=BATCHSTATUS,BATCHPROCESSORSTATUS,BATCHLISTCOUNT,EQUIPMENTCOUNT,TAGLISTCOUNT,TAGAELISTCOUNT,EVENTREADCOUNT,ERRORCOUNT,SOURCEUNITCOUNT,PIUNITCOUNT,SOURCEPHASEMODCOUNT,PIPHASEMODCOUNT,SOURCEBATCHCOUNT,PIBATCHCOUNT,SOURCEUNITBATCHCOUNT,PIUNITBATCHCOUNT,SOURCESUBBATCHCOUNT,PISUBBATCHCOUNT,SOURCEPROPERTYNODECOUNT,PIPROPERTYNODECOUNT,SOURCEPROPERTYEVENTCOUNT,PIPROPERTYEVENTCOUNT,SOURCETAGCOUNT,PITAGCOUNT,SOURCETAGEVENTCOUNT,PITAGEVENTCOUNT,SOURCETAGALIASCOUNT,PITAGALIASCOUNT,CACHEDBATCHCOUNT,OPENBATCHCOUNT,SOURCEREADTIME,TAGCACHETIME,BATCHCACHETIME,EQUIPMENTCACHETIME,BATCHSYNCTIME,TAGSYNCTIME,EQUIPMENTSYNCTIME,TOTALTIME,WAITINGFOREQUIPMENTUB,NUMBEROFWORKORDERS,NUMBEROFOPENWORKORDERS,WORKORDERQUERYTIMEAVG
```

For more information about each tag available, see Monitor tag reference.

Additionally, you can enable or disable all tags using the ALL or NONE options:

```
/enabledmonitortags=ALL
/enabledmonitortags=NONE
```

/equipmentXML =<filepath>

(Optional) Specifies the location of the DeltaV-generated equipment hierarchy XML file. The EMDVB interface uses this reference data to locate missing ProcessCell field by searching based on the combination of Area and Unit fields. Valid only when a DeltaV AE SQL datasource is defined.

Example:

```
/EquipmentXML="C:\DeltaV\Equip.xml"
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

/failoverID =<string>

(Optional) Configure a unique failover ID for the interface instance. Must be used with the /failover parameter.

Example: /FailOverID="intf1"

/failovertag =<tag name>

(Optional) Specifies the PI tag to be used to track the primary interface instance. Must be used with the /failoverID parameter.

Example: /failovertag="Batch_FailoverTag"

/host =host:port

(Required) Specifies the PI Data server or PI Collective where data is to be stored. Host is the IP address or domain name of the server node. Port is the port number for TCP/IP communication. The port is always 5450.

Examples:

- /host=marvin
- /host=marvin:5450
- /host=206.79.198.30
- /host=206.79.198.30:5450

/id =<identifier>

(Required) Specify a one- to nine-number identifier for the interface instance. Assigned to the Location1 attribute of tags that the interface instance updates.

/inifile =<path>

(Optional) Override the default location and name for the initialization file. By default, the .ini file resides in the interface installation directory and has the same file name and the .bat file.

/link =<AF element path>

(Deprecated, see /readlink and /writelink.) Combine event frames from different interface instances. Useful when you have an MES controlling multiple BESs. Configure an interface instance for each BES, specifying the same linkage element. The BES interface instances create event frame references under the MES event frames that refer to the BES interface instances. For Emerson Syncade systems, the AutomationBatchID field must match the batchID of the batch created by the BES.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

/maxqtf =<days>

(Optional) Sets the maximum number of days for which a query can return data. Used to break a large query into a set of smaller queries, to ensure that the system does not run out of memory. The value can be fractional.

- Minimum: 0.001
- Maximum: 180
- Default: 30

/maxstoptime =<seconds>

(Optional) Specifies (in seconds) the maximum time allowed for the interface to properly shutdown. If shutdown takes longer than the specified time, the interface is forced to terminate immediately. Default: 120 seconds

/merge

(Optional) Enable merging of multiple source batches with same ID. The original data for each batch merged is stored in PI properties under a node named using the ID of the original batch. The data includes the original batch ID, start time (UTC), end time (UTC), product and formula name. The interface merges only batches that are cached in local memory. The time frame for merging is configured using the `/cachetime` switch.

If the IDs of the batches you want to merge are different, use the `/bidm` flag to override incoming IDs.

Example: Given the following five batches running within the cache time frame.

- Test12345_1
- Test_12345_2
- CleaningTest
- USPO12345_test
- CleaningTest

With merging enabled, only the CleaningTest batches are merged. To merge the other three batches, which have IDs containing the string "12345", specify `/bidm=######`.

/mode =<mode>

(Optional) Valid modes are as follows:

- **Realtime:** (Default) Real-time data collection. If a recovery start time is specified (`/rst`), the interface recovers data before starting real-time collection.
- **Stat:** Statistics mode. Compare source data with the corresponding PI System batch data. The interface does not write to or modify any data PI batch data. On completion, the interface reports results and stops.
- **Delete:** Delete batch data from PI archives for specified period, leaving data from all other sources intact. Use only if the interface is unable to synchronize source batch data with the PI System. Must be used in conjunction with the recovery mode switches (`/rst` and `/ret`).

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- **NoData:** Newly-added tags, units and modules are indexed (referenced) in the primary PI archive, but older archives do not have entries for these modules, units and tags. In NoData mode, the interface creates modules, units, tags and tag aliases without processing batch data and writing events to the tags. To recover batch data for a period prior to the one in the primary archive, you must reprocess older archives with the offline archive utility. Manual archive reprocessing creates indexes for newly-added units, modules, tags. Always run the interface in this mode before writing new batch data to older PI archives (that is, archives other than the primary archive).

/monitortagwindow =<days>

(Optional) Monitor tag time window in days. Most monitor tags show a sum of events over a number of days. This setting allows the size of the window to be adjusted. The default value is one day.

/mop

(Optional) Merge identically named operations under the same parent unit procedure. The start time of the combined operation is the start of the earliest operation and the end time is the end time of the latest or longest operation that was merged.

/mup

(Optional) Merge identically named sequential unit procedures running on the same unit into a single unit procedure. Unit procedures are not merged if the unit was used by another recipe between candidates for merging. The start time of the resulting merged unit procedure is the start of the earliest unit procedure, and the end time is the end time of the latest or longest unit procedure that was merged.

/noarbitration

Create unit batches based solely on source batch recipe data. For use when the source Batch Executive System (BES) provides batch data without equipment arbitration.

/ns[=lang]

(Optional) Perform numerical conversions using the specified language's conventions. Useful when the numerical conventions differ from the default settings (for example, comma instead of decimal point). Default is "English_UnitedStates". If you omit the language parameter, the interface uses the "Regional and Language Options" settings in effect for the interface node.

Language types and abbreviations:

- chinese chinese-simplified (chs)
- chinese-traditional (cht)
- czech (csy)
- danish (dan)
- belgian, dutch-belgian (nlb)
- dutch (nld)

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- australian, english-aus (ena)
- canadian, english-can (enc)
- english english-nz (enz)
- english-uk (uk)
- american, american-english, english-american, english-us, english-usa, (enu) (us) (usa)
- finnish (fin)
- french-belgian (frb)
- french-canadian (frc)
- french (fra)
- french-swiss (frs)
- german-swiss, swiss (des)
- german (deu)
- german-austrian (dea)
- greek (ell)
- hungarian (hun)
- icelandic (isl)
- italian (ita)
- italian-swiss (its)
- japanese (jpn)
- korean (kor)
- norwegian-bokmal (nor)
- norwegian norwegian-nynorsk (non)
- polish (plk)
- portuguese-brazilian (ptb)
- portuguese (ptg)
- russian (rus)
- slovak (sky)
- spanish (esp)
- spanish-mexican (esm)
- spanish-modern (esn)
- swedish (sve)
- turkish (trk)

/piconnto =<seconds>

(Optional) Override the default SDK setting for PI connection timeout.

/pidato =<seconds>

(Optional) Override the default SDK setting for PI data access timeout.

/pipswd =<password>

(Optional) Specify the user password to be used to connect to the PI Data Archive. By default, the interface uses PI trusts for authentication.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

/piuser =<name>

(Optional) Specify the user name to be used to connect to the PI Data Archive. By default, the interface uses PI trusts for authentication.

/print =<filename>

(Optional) Prints the results of first scan to specified text file. The results include the batch tree, tag list, and equipment tree. Used for troubleshooting.

/ps =pointsource

Specifies the point source for the points maintained by the interface.

/ptsec =<string>

(Optional) Specifies the access security settings to be assigned to interface-generated tags. For PI Data Archive version 3.4.375.99 or earlier, use owner, group, world format.

Example:

```
/ptsec="o:rw g:r w:r"
```

For PI Data Archive version 3.4.380.36 or later, specify an access control list (ACL).

Example:

```
/ptsec="piadmin: A(r,w) | PIEngineers: A(r)"
```

/ras =<startstr, stopstr>

(Optional) Directs the interface to use the "Report" event to create phase steps under active phase states. The phase step name and start/stop events are obtained from the "Descript" column. The start and stop strings must start in the same position in the data source and must not be the first characters in the "Descript" column. The phase step name is derived from the characters preceding the start/stop text. Specify the strings in a double-quoted comma-separated list, as shown in the example below.

Example: `/ras="-STRT, -STOP"`

If the Descript Column contains TEST123-STRT-B, the interface generates a phase step named "TEST123" under the currently active phase state. Open phase steps are closed by the end of the parent operation and not by the end of parent phase or phase state.

/readlink= <AFelementpath>

Combine event frames from different interface instances. For an MES controlling one or more BES systems, configure `/readlink` on the MES interface and configure an interface instance for each BES, specifying the same linkage element in the BES `/writelink` setting. The BES interface instances will then create event frame references under the MES event frames that refer to the BES event frames. For Emerson Syncade systems, the AutomationBatchID field must match the batchID of the batch created by the BES.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

For a BES interface controlling one or more MES systems, configure `/readlink` on the MES interface and configure an interface instance for each BES, specifying the same linkage element in the BES `/writelink` setting. The MES interface will then create event frame references under the BES event frames that refer to the MES event frames. Link templates must also be configured to define which events specify a link.

/restef

(Optional) Enables an event frame with references to inherit security settings from its primary reference element.

/ret =<datetime>

(Optional) Specifies the end time for data recovery. The interface recovers batches that start before the specified time, including batches that end after the specified end time. Specify the time using the interface node format and time zone.

/retry =<seconds>

(Optional) Specifies how often the interface retries a failed attempt to write data. The default is 60 seconds.

/retryto =<seconds>

Specifies how long the interface retries a failed attempt to write data before timing out. By default, the interface never times out. If you configure a timeout setting, be advised that you risk losing data.

/rst =<datetime>

(Optional) Specifies recovery start time. The interface recovers batches that start after the specified time, as well as batches that start before the specified time but end after it. Specify the time using the interface node format and time zone.

/rti

Remove trailing index from Recipe fields. Applicable to Procedure, Unit Procedure, and Operation Recipe fields. Emerson EVT data source only.

/scan =<seconds>

(Optional) Specifies, in seconds, how often to scan the data source for new data. The default is 60 seconds. A scan that returns a large amount of data can cause the interface to skip the subsequent scan.

/singlerun

(Optional) Perform one scan and stop.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

/smp ="equipment path"

(Optional) Specifies an alternate PI Module path or PI AF element path for a particular equipment hierarchy. By default, the interface scans starting at the root level. Use the following syntax:

`\\<RootModule>\<SubModule>\<…>`

/sqlconnto =<seconds> (DeltaV SQL only)

(Optional) Override the default SQL timeout setting (60 seconds).

/sqldato=<seconds> (DeltaV SQL only)

(Optional) Override the default SQL data access timeout setting (100 seconds).

/swaptime =<seconds>

(Optional) Specifies, in seconds, how long the current primary interface must be unavailable before failover occurs. Default: 300 seconds.

/tbid

(Optional) Use the truncated batch ID in the batch ID field of unit procedures. Incoming batch IDs are reformatted using the mask defined in the `/bidm` parameter.

/tbse

(Optional) True Batch Start End (/TBSE) is a mechanism used to determine when the PI Batch initially started. /TBSE is also referred to as the recipe-based start/end or the true start and end time and is set to FALSE by default. This tool is helpful if you have reporting software, such as rtreports, and require a precise start time in your reports. It is intended for batches with S88 recipe types: Procedure, Unit Procedure, Operation, and Phase.

When /TBSE = `true` in the .ini configuration file, the interface uses top level recipe start/end events for creating batch objects. When processing a batch, you begin first with a batch event and followed by a procedure event. The interface uses the start time of the initial batch event prior to the procedure start event to indicate the start of the batch.

When a batch event ends, usually through a BATCH STOP event, the PROCEDURE COMPLETE State Change event timestamp lists the specific end time.

Note: We do not recommend configuring both the /TBSE and the Use Batch Recipe (/UBR) command line parameters to TRUE at the same time. These parameters are independent from one another and are best utilized when run in separate configurations.

If both /TBSE and /UBR are set to TRUE, it is likely that /TBSE will overwrite the start and end times of the batch.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

/ts=GMT | LCL

(Optional) Specifies how the interface interprets event timestamps from an SQL data source. Options are local time or GMT. Default is GMT.

/ubr

Use Batch Recipe (/UBR) is a mechanism that allows the interface to enable or disable the use of the batch recipe view to generate PI Batch database objects.

You can configure /UBR in the .ini configuration file, on the command line, or on the Batch Setup tab in the PI Event Frames Interface Manager. By default, /UBR is set to false in the Emerson batch interfaces.

When /UBR = false, the interface uses STATE CHANGE logic to control the start and end of event frames. Example State Change messages are RUNNING, REMOVED, ABORTED, COMPLETE, STOPPED and ABANDON. The interface combines the state change with the recipe (Batch, UnitProcedure, Operation, or Phase) to determine which recipe step has changed state.

When /UBR = true, the interface uses the Batch Recipe logic as opposed to the STATE CHANGE logic. The interface uses SYSTEM MESSAGE to control the start and end of event frames. Example system messages are BEGIN OF BATCH, END OF BATCH, UNIT PROCEDURE STARTED, and UNIT PROCEDURE ENDED.

Additionally, when UBR=TRUE, the start time for the queried batch events is the ACTIVATETIME instead of the STARTTIME. The end time for the batch events the DEACTIVATETIME instead of ENDTIME.

Note: We do not recommend configuring both the UBR and the True Batch Start End (TBSE) command line parameters to TRUE at the same time. These parameters are independent from one another and are best utilized when run in separate configurations.

If both /TBSE and /UBR are set to TRUE, it is likely that /TBSE will overwrite the start and end times of the batch.

/uidlist=<list>

(Optional) Recover specified manufacturing orders, then exit. Specify a comma-separated list of unique IDs for the manufacturing orders to be recovered. This parameter overrides any settings specified for the /rst and /ret recovery switches.

Example: /uidlist=MOKey#1, MOKey#2

/uobev (DeltaV SQL 9.3+ only)

(Optional) Directs the interface to use the original batch event view. By default the interface queries 17 tables to retrieve data for batch-associated events. Note that this view does not provide explicit [Descript], [Pval] and [EU] fields. Instead the [Descript] field combines data from all three fields. This option is provided for backward compatibility.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Initialization file reference

The initialization (.ini) file stores configuration information for an interface instance. The PI Event Frame Interface Manager configuration tool generates and updates this file each time you edit the instance configuration. You can access initialization files from `%ProgramData%\OSIsoft\Interfaces\`.

Note: Before release 4.x, initialization files were stored in the interface installation directory.

Tip: To ensure a valid .ini file, use the PI Event Frame Interface Manager configuration tool. Do not edit the .ini file manually.

Use this appendix to learn about the contents of the .ini file for troubleshooting purposes.

The .ini file contains the following sections:

- **COMMAND LINE PARAMETERS:** Defines basic interface settings.
- **SIMPLE SWITCHES:** Defines connection settings.
- **SOURCE TEMPLATE:** Defines the settings required to connect to the data source and events to be skipped (filtered).
- **RECIPE TEMPLATE:** Defines how data is recorded at each level of the batch hierarchy.
- **TAG TEMPLATE:** Defines how tags are created and updated based on incoming data from the data source.
- **PROPERTY TEMPLATE:** Define how PI properties or event frame attributes are created based on incoming data from the data source.
- **TRANSLATIONS:** Maps text from the data source language to the target language for the PI System.
- **DCS LINK TEMPLATE:** Define rules for linking event frames created by different interfaces.

In defining settings for templates, you can use the following wildcards to compose expressions:

Wildcard	Description
#	single digit numerical value (0-9)
@	single alphabetic character (a-z, A-Z)
?	any single character
!	repeat previous mask symbol
*	any number of characters

Source templates

A single interface instance can gather data from one or more data sources. In the .ini file, each data source is assigned an index, which is used to specify the settings for the data source. To specify a setting for a data source, use the following syntax:

```
Source[<index>].<setting> = <value>
```

Following are some simple examples of data source templates.

Single DeltaV Batch Historian:

```
Source[1].sqlserver = deltav10 Source[1].database = DVHisDB
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

DeltaV Batch Historian plus Alarms and Events:

```
Source[1].sqlserver = deltav10 Source[1].database = DVHisDB Source[2].sqls
erver = deltav10\DELTAV_CHRONICLE Source[2].database = Ejournal Source[2].
isAE = true
```

Data source template parameters

The following headings describe the parameters for data source templates.

cursor= client | server

Optional for SQL data source. Available in DeltaV 9.3+.

- Client (default): The interface retrieves complete dataset prior to processing. High memory consumption on interface node.
- Server: The interface requests and processes one dataset record at a time. Reduces interface node memory consumption.

evtdir=<path>

Required for event file data sources. This parameters defines the folder that contains the event files.

Example:

```
Source[1].evtdir = D:\TEST\RELEASE\test_1
```

excludestates=<list>

(Optional) Specifies a comma-separated list of phase states to ignore. Not case-sensitive. You can use wildcards for matching.

Examples:

```
excludestates=COMPLETED,AB*ING,IDLING, COMPLE*
```

isAE=true

Indicates that the data source is a DeltaV Alarms and Events server.

opcnode=<node name or IP address>

Required for OPC alarms and events data source. Available in DeltaV 10.3 and higher. Specifies the host of the DeltaV OPCAE server is installed. If used with DeltaV SQL server, must be defined for the same source.

Example:

```
Source[1].sqlserver = deltav10
Source[1].sqldatabase = DVHisDB
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
Source[1].opcnode = deltav10
Source[1].opcserver = DeltaV.OPCEventServer.1
```

opcserver=<server name>

Optional for OPC alarms and events data source. Specifies the name of the alarms and events server, if you are not using the default server.

skipphases=<list>

(Optional) Specifies a comma-separated list of phases to ignore. Not case-sensitive. You can use wildcards for matching. The interface checks the list against the [Phase] field (batch recipe) or [PhaseModule] field (equipment).

Examples:

```
skipphases=phase_1, ph*2,ph_test*, ph*ort*
```

skiprecipes=<list>

(Optional) Specifies a comma-separated list of recipes to ignore. Not case-sensitive. You can use wildcards for matching. The interface checks the list against the corresponding event field, depending on recipe type, as follows:

```
skiprecipes=recipe_1, rec*2,PRC_PAINT*, UP_TEST:2
```

skipunits=<list>

(Optional) Specifies a comma-separated list of units to ignore. Not case-sensitive. You can use wildcards for matching. The interface checks the list against the corresponding [Unit] field.

Example:

```
skipunits = unit_1, u*2
```

sqlserver=<node name or IP address>

Required for SQL Server data source. Specifies the host where SQL Server is running. Available in DeltaV 9.3+.

sqldatabase=<database name>

Optional for SQL Server data source. Specifies the name of the database, if you are not using the default database. (DVHisDB).

sqlpswd=<password>

For explicit login to SQL Server. Enter the password for the `sqluser` parameter.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

sqluser=<user name>

For explicit login to SQL Server. Use this parameter with `sqlpswd`.

Note: By default the interface uses Windows authentication to connect to SQL Server.

Recipe templates

To define the data generated for each level of the batch hierarchy, you define recipe templates.

Following is a typical set of default recipe templates, though the precise set of default templates that a batch interface provides for a BES is vendor-specific.

```
RECIPE[1].NAME=[PROCEDURE]
RECIPE[1].CATEGORY=OSIBatch
RECIPE[1].TEMPLATE=Procedure
RECIPE[1].BATCHID=[BATCHID]
RECIPE[1].PRODUCT=[PVAL]
RECIPE[1].PRODUCTTRIGGER=[EVENT, value="Recipe Header"][DESCRIPT, value="P
roduct Code"]
RECIPE[1].PROPERTY[1].NAME=[Descript]
RECIPE[1].PROPERTY[1].VALUE=[Pval]
RECIPE[1].PROPERTY[1].ENGUNITS=[EU]
RECIPE[1].PROPERTY[1].CATEGORY=Recipe Header
RECIPE[1].PROPERTY[1].TRIGGER=[Event, value="Recipe Header"] RECIPE[1].PRO
PERTY[2].NAME=[Descript]
RECIPE[1].PROPERTY[2].VALUE=[Pval]
RECIPE[1].PROPERTY[2].CATEGORY=Formula Header
RECIPE[1].PROPERTY[2].TRIGGER=[Event, value="Formula Header"]

RECIPE[2].NAME=[UNITPROCEDURE]
RECIPE[2].CATEGORY=OSIBatch
RECIPE[2].TEMPLATE=UnitProcedure
RECIPE[2].BATCHID=[BATCHID]
RECIPE[2].MODULEPATH=[AREA]\[PROCESSCELL]\[UNIT]
RECIPE[2].PRODUCT=[PVAL]
RECIPE[2].PRODUCTTRIGGER=[EVENT, value="Recipe Header"][DESCRIPT, value="P
roduct Code"]

RECIPE[3].NAME=[OPERATION]
RECIPE[3].CATEGORY=OSIBatch
RECIPE[3].TEMPLATE=Operation

RECIPE[4].NAME=[PHASE]
RECIPE[4].CATEGORY=OSIBatch
RECIPE[4].TEMPLATE=Phase
RECIPE[4].MODULEPATH=[AREA]\[PROCESSCELL]\[UNIT]\[PHASEMODULE]
RECIPE[5].NAME=[PHASESTATE]
RECIPE[5].CATEGORY=OSIBatch
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
RECIPE[5].TEMPLATE=PhaseState
RECIPE[6].NAME=[PHASESTEP]
RECIPE[6].CATEGORY=OSIBatch
RECIPE[6].TEMPLATE=PhaseStep
```

Tag templates

Tag templates specify how the interface creates and updates PI tags based on incoming data. For example, the following template creates one tag per unit for state change events:

```
Tag[1].Name=BESName:[UNIT].Event.[EVENT]
Tag[1].Descriptor=[UNIT] [EVENT]
Tag[1].EngUnits=[EU]
Tag[1].Value=[DESCRIPT]|[EVENT]|[PVAL]|[EU]|[AREA]|[PROCESSCELL]|[UNIT]|[P
HASEMODULE]|[USERID]
Tag[1].Type=string Tag[1].UnitAlias=Event.[EVENT]
Tag[1].Trigger=[EVENT,value="State Change"]
```

Tag template settings

The following table lists valid settings for tag templates. The timestamp for tag events is taken from the triggering event.

ANNOTATION

Annotate the tag using a string.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

ANNOTATION2

Annotate the tag using a name/value object.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

DESCRIPTOR

Specifies how the descriptor field of the tag is populated.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

EU

Specifies the engineering units for the tag.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

NAME

Specifies how the tag is to be named.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

PHASEALIAS

Configures the naming convention for the phase module alias generated by the interface. The unit alias refers to the AF asset representing the phase module associated with the incoming event.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [@,value="Exact Field"]
- [#,value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

TRIGGER

Specifies the event that causes the interface to generate or update the tag. To define a trigger, specify an expression composed of a placeholder and value. When the interface detects the specified value in the placeholder, it generates or updates the tag.

You can specify multiple triggers for a single tag. If you specify the triggers on a single line, the tag is generated only when all the conditions are met (logical AND). If you specify the trigger expressions on separate lines, the tag is generated when any of the conditions is met (logical OR).

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

TRANSLATE

To translate the value according to the translation map that you define, set to TRUE.

Valid Placeholders

- TRUE

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- FALSE

TYPE

Specifies the data type for the tag. To configure the interface to evaluate the data and assign the data type accordingly, specify AUTO.

Valid Placeholders

- STRING
- FLOAT
- INTEGER
- AUTO

UNITALIAS

Configures the naming convention for the unit alias generated by the interface. The unit alias refers to the AF asset representing the unit associated with the incoming event.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

VALUE

(Required) Specifies the value to be stored in the tag. To compose the value, you can use free text plus valid placeholders.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [TAG]
- [TIME]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

Property templates

To define properties (batch database) or attributes (PI AF) to be stored at a particular level of the batch hierarchy, you create property templates. The syntax for property templates is as follows:

```
PROPERTY[<index>].<setting> = <value>
```

Instead of PROPERTY, you can specify ATTRIBUTE. The `<index>` is a unique identifier used to group the settings for a particular property. To include data from the data source, specify placeholders.

For example, if you define the following template:

```
Property[1].Value=[Time]:[Descript]:[Pval]:[EU]-[Event]_Testing
```

...and the interface applies it to the following incoming data:

```
[Time]="12/01/2015 12:01:05" [Descript]=abc [Pval]=123.456 [EU]=cm [Event
]=Report
```

The following property is added to the resulting batch or batch-level event frame:

```
12/01/2015 12:01:05:abc:123.456:cm-Report_Testing
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Property template settings

The following headings describe the settings required to define property templates.

NAME

(Optional) Specifies how the property is to be named. You can use an expression to configure the name, but note that the resulting name must be unique. If two different name templates generate the same name, the property value is overwritten and no error is logged. If you omit the name setting, the interface assigns a default name ("Event_n").

By default, properties are created at the level of the batch hierarchy that triggers their creation. To create a property at the top of the hierarchy, regardless of where it was triggered, specify "$" at the beginning of the expression that defines the name.

For example:

```
Property[1].Name = $\[Event]
```

Additionally, you can use the not equals syntax ("!+") with anything that has a comparison.

For example:

```
Property[1].Trigger = [UNITPROCEDURE, value!=""]
Property[1].Trigger = [UNITPROCEDURE, value!="*"]
Property[1].Trigger = [UNITPROCEDURE, value!="aprefix"]
```

In this example, the property on the first UNITPROCEDURE placeholder triggers without meeting any condition. The second placeholder triggers with any value while the third triggers only if the condition equals "aprefix".

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PHASESTATE]
- [PHASESTEP]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [TIME]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

VALUE

(Required) Specifies the value to be stored in the property. To compose the value, you can use free text plus valid placeholders.

Basic example:

```
Property[1].Value = [BatchID] | event: <State*> | [Descript] | val: [Pval]
```

Advanced parsing:

```
Property[1].Value = [BatchID] | event: [*,value="State*"] | [Descript] | v
al: [Pval]
```

Note: For SQL data sources with the "Use original batch event view" option enabled (/UOBEV), you cannot use the [PVAL] or [EU] placeholders. To obtain this data you must parse it from the [DESCRIPT] placeholder.

Additionally, you can use the not equals syntax ("!+") with anything that has a comparison.

For example:

```
Property[1].Name = $\[Event, value!=""]
Property[1].Name = $\[Operation, value!=""]
```

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PHASESTATE]
- [PHASESTEP]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [TAG]
- [TIME]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

TRIGGER

Specifies the event that causes the interface to generate the property. To define a trigger, specify an expression composed of a placeholder and value. When the interface detects the specified value in the placeholder, it generates the property.

Examples:

```
Property[1].Trigger = [Parameter, value="State Ch*"]
Property[1].Trigger = [Value, value="tes*"]
Property[1].Trigger=[Event, value="State*] [Pval,value=RUNNING"]
Property[1].Trigger=[Event]
```

Additionally, you can use the not equals syntax ("!+") with anything that has a comparison.

For example:

```
Property[1].Trigger = [UNITPROCEDURE, value!=""]
Property[1].Trigger = [UNITPROCEDURE, value!="*"]
Property[1].Trigger = [UNITPROCEDURE, value!="aprefix"]
```

In this example, the property on the first UNITPROCEDURE placeholder triggers without meeting any condition. The second placeholder triggers with any value while the third triggers only if the condition equals "aprefix".

You can specify multiple triggers for a single property. If you specify the triggers on a single line, the property is generated only when all the conditions are met (logical AND). If you specify the trigger expressions on separate lines, the property is generated when any of the conditions is met (logical OR).

Here is an example of a **logical AND** trigger where all conditions must be met:

```
.TRIGGER=[Event, value="Recipe Header"][Descript, value="Description"][Val
ue, value="My Description"]
```

Conversely, the following is an example of a **logical OR** trigger. At least one of these conditions must be met:

```
.TRIGGER=[COMMENT, value=""]
.TRIGGER=[EVENT, value=three]
.TRIGGER=[PHASESTATE, value=active]
.TRIGGER=[UNIT, value=S88]
```

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [PHASEMODULE]
- [PHASESTATE]
- [PHASESTEP]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

TRANSLATE

Specifies values for translation according to the translation map you define. Values set to TRUE are translated.

Valid Placeholders

- TRUE
- FALSE

TYPE

Specifies the data type for the value. To configure the interface to evaluate the data and assign the data type accordingly, specify **AUTO**.

Valid Placeholders

- STRING
- FLOAT
- INTEGER
- AUTO

CATEGORY

(AF only) Specifies the AF category to be assigned to the attribute.

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PHASESTATE]
- [PHASESTEP]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [TIME]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]
- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

UOM or ENGUNITS or EU

Specifies the units of measure for the attribute. (See /UOBEV note for VALUE setting, above.)

Valid Placeholders

- [AREA]
- [BATCHID]
- [COMMENT]
- [DESCRIPT]
- [EU]
- [EVENT]
- [OPERATION]
- [PHASE]
- [PHASEMODULE]
- [PHASESTATE]
- [PHASESTEP]
- [PROCEDURE]
- [PROCESSCELL]
- [PVAL]
- [TIME]
- [UNIQUEID]
- [UNIT]
- [UNITPROCEDURE]
- [USERID]
- [@.value="Exact Field"]

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

- [#.value="Field Mask"]
- [!.value="Example"]
- [?.value="Tag"]
- [*.value="Unit"]
- Advanced parsing

Translation templates

Following is a simple template that translates German source data to English for storage in the PI System.

```
TRANSLATE: "Grundrezept" = "Procedure"
TRANSLATE: "Teilrezept" = "Unit Procedure"
TRANSLATE: "Grundoperation" = "Operation"
TRANSLATE: "Grundfunktion" = "Phase"
```

Initialization file examples

Multiple EVT Sources

```
[SOURCE TEMPLATE]

Source[1].evtdir=c:\test\evt
Source[2].evtdir=\\deltav9\\journals\evt

[GENERAL]

Excludestates=COMPLETE, ABORTING Equipment = abs:[Unit]\[PhaseModule]\Misc


[TAG TEMPLATE]

// [Basic Tag template, triggered on Event=Report, aliases are created as
tag name]

Tag[1].Name = [Unit]_[PhaseModule]_Report
Tag[1].Value = [Pval]
Tag[1].Type = Float // [Tag template with custom aliases, triggered on Eve
nt=Owner Change]
Tag[2].Name = [Unit]_[PhaseModule]_Owner Change
Tag[2].Value = [time]_[Descript]
Tag[2].Type = String
Tag[2].unitalias = [PhaseModule] Owner Change Me
Tag[2].phasealias = Owner Change Me

// [Tag template with custom aliases, triggered on set of events defined as
 triggers]
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
// [Note: Unitalias and Phasealias are NOT going to be created since there
 are no Unit or Phase Module

// defined in the tag name]

Tag[3].Name = Generic Tag
Tag[3].Value = [time]_[Event]_[BatchID]_[pval]
Tag[3].Type=String
Tag[3].trigger = Report
Tag[3].trigger = Owner Change
Tag[3].trigger = Operator Prompt
Tag[3].unitalias = [phasemodule] abcd
Tag[3].phasealias = testing [PROPERTY TEMPLATE] Property[1].Value = [Time]
 State Change [Descript] [pval]
```

DeltaV German EVT Source

```
[SOURCE TEMPLATE]

source[1].evtdir = D:\TEST\evt german\evt

[GENERAL]

ExcludeStates = NONE

[TAG TEMPLATE]
tag[1].Name = German Report
tag[1].Value = [Descript]:[event][pVal]
tag[1].Type = string
tag[2].Name = German Bericht
tag[2].Value = [Descript]:[event][pVal]
tag[2].Type = string
tag[3].Name = German Bericht float
tag[3].Value = [pVal]
tag[3].Type = float

[PROPERTY TEMPLATE]

Property[1].Value = [Time] Bericht [Unit] [phasemodule] [descript]-[Pval]
Property[2].Value = [Time] report [Unit] [phasemodule] [descript]-[Pval]

[TRANSLATIONS]

// [S88 Levels]
translate: "Grundrezept" = "Procedure"
translate: "Teilrezept" = "Unit Procedure"
translate: "Grundoperation" = "Operation"
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
translate: "Grundfunktion" = "Phase"

// [Batch Header info]

translate: "Rezeptkopf" = "Recipe Header"
translate: "Produktcode" = "Product Code"
translate: "Formelkopf" = "Formula Header"
translate: "Formelname" = "Formula Name"

// [Arbitrations]

translate: "Rezeptzuteilung" = "Recipe Arbitration"
translate: "Betriebsmittel belegt durch Rezept" = "Resource Acquired by re
cipe"
translate: "Betriebsmittel freigegeben durch Rezept" = "Resource Released
by recipe"
translate: "Teilanlage" = "Unit"

// [Recipe Logic, Comment needed only if there is an extra column in Delta
V, so event can be converted to
// Comment event]

translate: "Zustands\E4\nderung" = "State Change"
translate: "Kommentar" = "Comment"
translate: "Systemmeldung" = "System Message"
translate: "CHARGEN-Anfang" = "Beginning Of BATCH"
translate: "CHARGEN-Ende" = "End Of BATCH"
translate: "Teilrezept gestartet" = "Unit Procedure Started"
translate: "Teilrezept beendet" = "Unit Procedure Finished"
translate: "Grundoperation gestartet" = "Operation Started"
translate: "Grundoperation beendet" = "Operation Finished"

// [Phase States]

translate: "L\C4\UFT" = "RUNNING"
translate: "BEENDET" = "COMPLETE"
translate: "ENTFERNT" = "REMOVED"
translate: "GESTOPPT" = STOPPED" translate: "ABGEBROCHEN" = "ABORTED"

// [Additional Events to translate]

translate: "Bericht" = "Report"
translate: "Schrittaktivit\E4\t" = "Step Activity"
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

DeltaV SQL

```
[SOURCE TEMPLATE]

source[1].sqlserver = deltav10 source[1].sqldatabase = DVHisDB
source[2].sqlserver = deltav102 [GENERAL] ExcludeStates = NONE

[TAG TEMPLATE]

Tag[1].Name = [Unit] [Phasemodule] Report
Tag[1].Value = [Descript]
Tag[1].Type = string Tag[1].unitalias = [phasemodule] Report
Tag[1].phasealias = Report
Tag[2].Name = [Unit] [phasemodule] Tester
Tag[2].Value = [ atched] [Descript]=[event]
Tag[2].Type = string
Tag[2].Trigger = Owner Change
Tag[2].Trigger = Report
Tag[2].unitalias = [phasemodule] tester alias
Tag[2].phasealias = tester alias
```

DeltaV SQL, OPCAE

```
[SOURCE TEMPLATE]

source[1].opcnode = deltav101
source[1].opcserver = DeltaV.OPCEventServer.1 source[1].sqlserver = deltav
10
source[1].sqldatabase = DVHisDB [GENERAL] Equipment=Areas\Abs[Area]\Proces
sCells\sss_[ProcessCell]\sdf:[Unit]\Phases\[Phasemodule]_testing Product =
 [Product],Undefined SkipUnits = NULL*, D50* SkipPhases = Clean*, Load* Exc
ludeStates = IDLE, ABOR*G, STOP*G [TAG TEMPLATE]

// [DeltaV Tag Templates]

Tag[1].Name = [Unit] Report Tag[1].Value = [Descript] | [Pval] | [EU]
Tag[1].Trigger = [Event,value="Report"] Tag[1].Type = string
Tag[1].unitalias = NONE Tag[1].unitalias = NONE

// Multiple events triggering same template

Tag[2].Name = [Unit] [phasemodule] Tester
Tag[2].Value = [ atched] [Descript]_[event]
Tag[2].Type = string
Tag[2].Trigger = Owner Change
Tag[2].Trigger = Report
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

```
Tag[2].unitalias = [phasemodule] tester alias
Tag[2].phasealias = tester alias
```

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Diagnostic point reference

Performance tags, which are automatically created for each instance of the interface, enable you to monitor the performance of the instance. There are 35 performance tags, categorized as follows:

- Health monitoring
- Object counters
- Timers

Health monitoring tags

There are two tags designed to monitor the health of the interface: the HeartBeat tag and the DeviceStatus tag. The HeartBeat tag is updated with the scan frequency configured for the interface or 60 seconds, whichever is less. The value of the HeartBeat tag is a cycle of integers from 1 to 15. The device status tag is automatically created and configured by the interface on startup. The device tag reflects status as follows:

- `Good`: The interface is properly communicating and reading data from the data sources.
- `1 | Starting`: The interface is executing its initialization routines.
- `2 | <details>`: Indicates successful connection to the data source.
- `3 | <details>`: Indicates failure to access the event journal file directory or failure to read data from the event journal file.
- `4 | Shutdown`: The interface is shutting down.

The properties of health monitoring tags are provided in the table below, where the *Prefix* represents ``_.

Note: If these tags do not exist, the interface creates them automatically during startup.

Tag name	Point Type	Loc1	Loc3	PointSource	ExcDesc
Prefix_DeviceStatus	Int32	Intf ID	0	Intf Pt Src	[UI_DEVSTAT]
Prefix_HeartBeat	Int32	Intf ID	1	Intf Pt Src	[UI_HEARTBEAT]

Object counter tags

The first time it is started, the interface creates 24 tags designed to monitor its performance. Archiving flag for these tags is turned off. The following table contains common point attributes for this group of tags:

Point type	Location1	Point source
Int32	`<Interface ID>`	`<Interface Point Source>`

The attributes for each performance counter tag are provided in the table below where the Prefix is defined as _. All counters are set to 0 when the interface starts up.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Tag Name	Loc3	ExcDesc	Description
<interfaceID>_EventReadCount	2	[UI_EVENTREADCOUNT]	Number of events read from the source since startup.
<interfaceID>_ErrorCount	3	[UI_ERRORCOUNT]	Number of errors since startup.
<interfaceID>_SourceUnitCount	4	[UI_SOURCEUNITCOUNT]	Number of units found on the data source(s) since startup.
<interfaceID>_PIUnitCount	5	[UI_PIUNITCOUNT]	Number of units added since startup.
<interfaceID>_SourcePhaseModCount	6	[UI_SOURCEPHASEMODCOUNT]	Number of phase modules found on the data source(s) since startup.
<interfaceID>_PIPhaseModCount	7	[UI_PIPHASEMODCOUNT]	Number of phase modules added since startup.
<interfaceID>_SourceBatchCount	8	[UI_SOURCEBATCHCOUNT]	Number of batches found on the data source(s) since startup.
<interfaceID>_PIBatchCount	9	[UI_PIBATCHCOUNT]	Number of batches added since startup.
<interfaceID>_SourceUnitBatchCount	10	[UI_SOURCEUNITBATCHCOUNT]	Number of unit batches found on the data sources(s) since startup.
<interfaceID>_PIUnitBatchCount	11	[UI_PIUNITBATCHCOUNT]	Number of unit batches added since startup.
<interfaceID>_SourceSubBatchCount	12	[UI_SOURCESUBBATCHCOUNT]	Total number of operations, phases, and phase states found on the data source since startup.
<interfaceID>_PISubBatchCount	13	[UI_PISUBBATCHCOUNT]	Total number of operations, phases, and phase states added since startup.
<interfaceID>_SourcePropertyNodeCount	14	[UI_SOURCEPROPNODECOUNT]	Number of property nodes found in data source(s) since startup.
<interfaceID>_PIPropertyNodeCount	15	[UI_PIPROPNODECOUNT]	Number of PIProperty objects (nodes) added since startup.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Tag Name	Loc3	ExcDesc	Description
<interfaceID>_SourcePropertyEventCount	16	[UI_SOURCEPROPEVENTCOUNT]	Number of events to be written to the batch properties found on the data source(s) since startup.
<interfaceID>_PIPropertyEventCount	17	[UI_PIPROPEVENTCOUNT]	Number of PIProperties (events) added since startup.
<interfaceID>_SourceTagCount	18	[UI_SOURCETAGCOUNT]	Number of tags found on the data source(s) since startup
<interfaceID>_PITagCount	19	[UI_PITAGCOUNT]	Number of PI tags added since startup.
<interfaceID>_SourceTagEventCount	20	[UI_SOURCETAGEVENTCOUNT]	Number of events to be written into tags found on the data sources(s) since startup.
<interfaceID>_PITagEventCount	21	[UI_PITAGEVENTCOUNT]	Number of events written into PI points since startup.
<interfaceID>_SourceTagAliasCount	22	[UI_SOURCETAGALIASCOUNT]	Number of tag aliases to be created based on the data source(s) since startup.
<interfaceID>_PITagAliasCount	23	[UI_PITAGALIASCOUNT]	Number of PI aliases added since startup.
<interfaceID>_CachedBatchCount	24	[UI_CACHEDBATCHCOUNT]	Number of batch objects cached in the local memory.
<interfaceID>_OpenBatchCount	25	[UI_OPENBATCHCOUNT]	Subset of cached objects that have no end time set.
<interfaceID>_WaitingForEquipmentUB	34	[UI_UBWAITFOREQUIP]	Number of unit batches that do not have equipment allocated yet.

Timers

The last performance tag category is composed of timer tags, which are built automatically on first interface startup. Each timer tag reports on how much time per scan it took the interface to perform a particular task. There are three categories of timer tag: data source reading, local data caching and synchronizing cached data with the PI System.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

TagName	Loc3	ExcDesc	Description
<interfaceID>_SourceReadTime	26	[UI_SOURCEREADTIME]	The time per scan it took the interface to read data from data source(s).
<interfaceID>_TagCacheTime	27	[UI_TAGCACHETIME]	The time per scan it took the interface to populate local tag cache.
<interfaceID>_BatchCacheTime	28	[UI_BATCHCACHETIME]	The time per scan it took the interface to populate the local batch cache.
<interfaceID>_EquipmentCacheTime	29	[UI_EQUIPCACHETIME]	The time per scan it took the interface to populate the local equipment (module) cache.
<interfaceID>_BatchSyncTime	30	[UI_BATCHSYNCTIME]	The time per scan it took the interface to synchronize local batch cache with the PI System.
<interfaceID>_TagSyncTime	31	[UI_TAGSYNCTIME]	The time per scan it took the interface to synchronize local tag cache with the PI System.
<interfaceID>_EquipmentSyncTime	32	[UI_EQUIPSYNCTIME]	The time per scan it took the interface to synchronize local equipment cache with the PI System.
<interfaceID>_TotalTime	33	[UI_TOTALTIME]	The total time per scan it took to read data, cache it in local memory and synchronize the local cache with the PI System.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Glossary

Available terms

The following are high-context terminologies used in this document.

- Buffering
- N-Way Buffering
- ICU
- ICU Control
- Interface Node
- PI API
- PI Data Archive Collective
- PI Data Archive Node
- PIHOME
- PIHOME64
- PI Message Log
- PI Server Node
- PI SMT
- Pipc.log
- Point
- Service
- Tag (Input Tag and Output Tag)

Buffering

Buffering refers to an interface node's ability to store temporarily the data that interfaces collect and to forward these data to the appropriate PI Data Archives.

N-Way Buffering

If you have PI Data Archives that are part of a collective, PIBufss supports n-way buffering. N-way buffering refers to the ability of a buffering application to send the same data to each of the PI Data Archives in a collective. (Bufserv also supports n-way buffering to multiple PI Data Archives in a collective; however, it does not guarantee identical archive records since point compression attributes could be different between PI Data Archives. With this in mind, OSIsoft recommends that you run PIBufss instead.)

ICU

ICU refers to the PI Interface Configuration Utility. The ICU is the primary application that you use to configure PI interface programs. You must install the ICU on the same computer on which an interface runs. A single copy of the ICU manages all of the interfaces on a particular computer. You can configure an interface by editing a

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

startup command file. However, OSIsoft discourages this approach. Instead, OSIsoft strongly recommends that you use the ICU for interface management tasks.

ICU Control

An ICU control is a plug-in to the ICU. Whereas the ICU handles functionality common to all interfaces, an ICU control implements interface-specific behavior. Most PI interfaces have an associated ICU control.

Interface Node

An interface node is a computer on which PI Data Archive programs are not installed.

PI API

The PI API is a library of functions that allow applications to communicate and exchange data with the PI Data Archive. All PI interfaces use the PI API.

PI Data Archive Collective

A PI Data Archive Collective is two or more replicated PI Data Archives that collect data concurrently. Collectives are part of the High Availability environment. When the primary PI Data Archive in a collective becomes unavailable, a secondary collective member node seamlessly continues to collect and provide data access to your PI clients.

PI Data Archive Node

A PI Data Archive node is a computer on which PI Data Archive programs are installed. The PI Data Archive runs on the PI Data Archive node. In earlier documentation, PI Data Archive was referred to as the PI Server.

PIHOME

PIHOME refers to the directory that is the common location for PI 32-bit client applications. A typical PIHOME on a 32-bit operating system is: C:\Program Files\PIPC. A typical PIHOME on a 64-bit operating system is: C:\Program Files (x86)\PIPC. PI 32-bit interfaces reside in a subdirectory of the Interfaces directory under PIHOME. For example, files for the 32-bit Modbus Ethernet Interface are in: [PIHOME]\PIPC\Interfaces\ModbusE. This document uses [PIHOME] as an abbreviation for the complete PIHOME or PIHOME64 directory path. For example, ICU files in: [PIHOME]\ICU.

PIHOME64

PIHOME64 is found only on a 64-bit operating system and refers to the directory that is the common location for PI 64-bit client applications. A typical PIHOME64 is: C:\Program Files\PIPC. PI 64-bit interfaces reside in a subdirectory of the Interfaces directory under PIHOME64. For example, files for a 64-bit Modbus Ethernet

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Interface would be found in: C:\Program Files\PIPC\Interfaces\ModbusE. This document uses [PIHOME] as an abbreviation for the complete PIHOME or PIHOME64 directory path. For example, ICU files in: [PIHOME]\ICU.

PI Message Log

The PI message log is the file to which OSIsoft interfaces (based on UniInt 4.5.0.x and later) write informational, debug and error messages. When a PI interface runs, it writes to the local PI message log. This message file can only be viewed using the PIGetMsg utility. See the Message Logs section of the PI Universal Interface (UniInt) User Guide for more information on how to access these messages.

PI Server Node

In earlier documentation, the term "PI Server" was used as a nickname for the PI Data Archive and a PI Server node was a computer on which PI Data Archive programs were installed. While the PI Data Archive remains a core server of the PI Server product, the product name "PI Server" now refers to much more than the PI Data Archive. OSIsoft documentation, including this user manual, is changing to use "PI Server" in this broader sense and "PI Data Archive" to refer to the historian core.

PI SMT

PI SMT refers to PI System Management Tools. PI SMT is the program that you use for configuring PI Data Archives. A single copy of PI SMT manages multiple PI Data Archives. PI SMT runs on either a PI Data Archive node or an interface node.

Pipc.log

The pipc.log file is the file to which OSIsoft interfaces based on UniInt versions earlier than 4.5.0.x write informational and error messages. When a PI interface runs, it writes to the pipc.log file. The ICU allows easy access to the pipc.log.

Point

The PI point is the basic building block for controlling data flow to and from the PI Data Archive. For a given timestamp, a PI point holds a single value. A PI point does not necessarily correspond to a "point" on the data source device. For example, a single "point" on the data source device can consist of a set point, a process value, an alarm limit, and a discrete value. These four pieces of information require four separate PI points.

Service

A Service is a Windows program that runs without user interaction. A service continues to run after you have logged off from Windows. It has the ability to start up when the computer itself starts up. The ICU allows you to configure a PI interface to run as a service.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Tag (Input Tag and Output Tag)

The Tag attribute of a PI point is the name of the PI point. There is a one-to-one correspondence between the name of a point and the point itself. Because of this relationship, PI System documentation uses the terms "tag" and "point" interchangeably. Interfaces read values from a device and write these values to an input tag. Interfaces use an output tag to write a value to the device.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Release Notes

PI Interface for Emerson DeltaV Batch

Version 5.1.0.16

Overview

PI Interface for Emerson DeltaV captures batch, manufacturing, and process data from an Emerson DeltaV System. The interface collects data from a DeltaV SQL server or EVT files, converts this data to PI tags and PI batch properties, and stores the collected data as event frames and elements within PI Asset Framework (PI AF). You can use tag templates to control PI point creation and event population. All modules, tags, tag aliases, and health tags are automatically created on the PI server.

The 5.1.0.16 release of PI Interface for Emerson DeltaV Batch internally replaces PI SDK with AF SDK - this improves the security, robustness, and communication capabilities of the interface. As a result, this interface only writes Event Frames to a PI AF Server, and disallows creating PI Batches in the PI Batch Database.

Users creating PI Batches in the PI Batch Database need to migrate to Event Frames in order to utilize this version of the interface.

Fixes and Enhancements

Enhancements

There are no enhancements for this release.

Fixes

This section lists items that were resolved or added in this release of PI Interface for Emerson Delta V.

Work Item	Description
22132	History recovery is fixed to prevent infinite loop scenario when querying for SQL data. In addition, fixed time as double to string time conversions There was a millisecond precision error converting the time stored as doubles to a time string.
22133	Attribute values referencing other event frames are captured regardless of the order event frames are created.
22140	Unirecord data sources are now included in unirecord imports and exports.
22153	Restore DESCRIPT handling in phase step logic.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Work Item	Description
22156	Cache start and end times can no longer be permanently altered by arbitration events.
22163	SQL Native client print message was disabled for Oracle based interfaces.
22171	Changed the Batch-list management algorithm to improve event processing accuracy (prevent missed batches).
22172	When removing an interface from PI Event Frames Interface Manager, it will also remove any associated services.
22181	When adding an interface to PI Event Frames Interface Manager, the instance ID will be initialized as the same value of interface ID.
22209	Interface will no longer have a delay in processing a batch when the local time is greater than GMT.
22211	MONITORTAGWINDOW now defaults to 0, meaning monitor tag counts don't get trimmed to the most recent days of activity and roll over at INT_MAX unless MONITORTAGWINDOW is deliberately set.
22216	Interface ID has been added to the message logs in order to uniquely identify log message.
22218	All generated unirecords will have the placeholder [TIMESTAMP] have the format "YYYY-MM-DD" for a SQL source and "YYYY/MM/DD" for non SQL source like EVT files.
22221	Monitor tags now accurately reflect the number of batches and their child events.
22224	Bifconfig properly creates new default virtual user service without errors.
22235	Interface configuration for RST and RET reads UnitedStates format standard.
22245	The unit procedure batch id attribute is controlled by the unit procedure recipe template batchid.
22246	BifConfig does not allow older interfaces to edit health tags.
22249	All batch interfaces remove passwords from INI file regardless of parameter capitalization.
22250	Merging issue causes unit procedure and operation not to close.
22260	Batch event frames no longer merge event frames when merging is disabled and unique ids don't match.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Work Item	Description
22268	All RST and RET within the configuration will be parsed as US format. Fix in current development branch and in 4.3.x.x Branch.
22275	A check that verifies that a Procedure Event Frame and a Unit Procedure Event Frame are both generated by the same interface instance before linking them together.
22283	The Event log source is created during the interface install step.
22464	For interfaces with a version 5.0.0 or greater, PI Event Frames Interface manager nows tail the windows event viewer instead of PI Message log.
23177	BifConfig now stores RST and RET as UTC Note that they can be edited in a local friendly time string and read into the interface correctly. When written by the configuration program, they will write the times out as UTC and a local time string as a comment.
23299	EMDVB fetches SQL data with greater efficiency. EMDVB gathers SQL data for lists of batches rather than querying each batch's data individually, and reuses the same SQL connection for all queries.
24781	The interface setup kit is using the same GUID for all the interface. This causes trouble when installing another batch interface on the same machine. Fixed to use different GUID for each interface.
24847	SQL Native Client is no longer supported, and is now replaced with OLEDB for all supported batch interfaces.
25811	PI data server collectives and PI buffering to multiple PI data servers is now supported.
26756	With the release of version 5, if you have a batch interface supported batch database, it will no longer be able to write to PIBatch Database. Version 5 only supports Event Frames Customers who need to write to PIBatch. The database needs to stay with Version 4.x.x.x Batch Interfaces.
50935	Addressed advanced parsing issue when using LBE ("Left Bound Exclusive").
52120	Whenever an interface becomes the primary, it will use the /RST command line switch as the start date and process the batches.
67289	UniRecord exports now include all UniRecord fields.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Work Item	Description
67298	Event sorting is faster and memory consumption is better managed during event queuing.
67926	Interface will not allow Batch events with a zero start time.
68207	BifConfig will now store SCAN values as low as one second without complaint.
68907	Default account to setup interface service is now Default Virtual User (NT Service/).
69287	Updated the PI Event Frames Interface Manager utility to only allow TagPath properties to be set on Asset Templates and Property Templates. Fixed issue where the TAGPATH value was not being saved properly.
69486	All attributes are now created on the correct event frame.
69854	Fixes minor failover issues and includes a health tag that can be used to monitor failover status.
70398	Float tag values are now evaluated correctly.
70403	Message logging includes exception data.
72522	Interface stability improvements.
73489	Interface no longer hangs due to locale thread lock issue.
73498	Setup kit creates the event source name under application logs to match with interface application name, which is used by logging framework.
75880	Procedure level recipe events are processed correctly and have no missing end times.
76997	Logging added in incomplete data situations such as true batch start not set, no name alias set, unit not set, and matching arbitration event not found.
77832	Source queries no longer fail when querying over one thousand batches at once.

Known Issues

There are no known issues for this release.

System Requirements

Operating Systems

This interface is a 64-bit application.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

Platform	64-bit Application
Windows Server 2022 (64-bit)	Yes
Windows Server 2019 (64-bit)	Yes
Windows Server 2016 (64-bit)	Yes
Windows 11 (64-bit)	Yes
Windows 10 (64-bit)	Yes

Distribution Kit Files

Product	Software Version
Microsoft Visual C++ 2015-2019 Redistributable (x86)	14.21.27702
Microsoft Visual C++ 2015-2019 Redistributable (x64)	14.21.27702
PI AF Client 2018 SP3 Patch 4	2.10.10.2539
PI Interface for Emerson DeltaV Batch (EMDVB)	5.1.0.16
PI Network Subsystem Support (PINS)*	3.4.435.538

*The PI Network Subsystem Support (PINS) component is not displayed on the installation welcome screen if the PI Data Archive is installed already.

Installation and Upgrade

The PI Interface for Emerson DeltaV Batch can be installed or upgraded using the PI Interface for Emerson DeltaV Batch installation kit, EMDVB_5.1.0.16_.exe. This installation kit can be obtained by using the How to Download Products link listed in the OSIsoft Customer Portal How To's list. This list is located on the OSIsoft Customer Portal.

For additional information regarding the PI Interface for Emerson DeltaV Batch installation, please see the Installation instructions portion of the PI Interface for Emerson DeltaV Batch User Guide. This user guide is available for download from the OSIsoft Customer Portal.

Uninstalling PI Interface for Emerson DeltaV Batch Interface

The PI Interface for Emerson DeltaV Batch Interface can be uninstalled using the *Programs and Features* list accessible from the *Windows Control Panel*. After accessing the Programs and Features list, select the entry named *PI Interface for Emerson DeltaV Batch (EMDVB)* and then select *Uninstall* from the menu.

Security information and guidance

We are committed to releasing secure products. This section is intended to provide relevant security-related information to guide your installation or upgrade decision.

We proactively disclose aggregate information about the number and severity of security vulnerabilities addressed in each release. The tables below provide an overview of security issues addressed and their relative severity based on standard scoring.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

No security-related information is applicable to this release.

There are no security vulnerabilities in this release.

©2023 AVEVA Group plc and its subsidiaries. All rights reserved.

AVEVA Group plc
High Cross
Madingley Road
Cambridge
CB3 0HB
UK

Tel +44 (0)1223 556655

www.aveva.com
To find your local AVEVA office, visit **www.aveva.com/offices**

AVEVA believes the information in this publication is correct as of its publication date. As part of continued product development, such information is subject to change without prior notice and is related to the current software release. AVEVA is not responsible for any inadvertent errors. All product names mentioned are the trademarks of their respective holders.